Mining Discriminative Patterns for Classifying Trajectories on Road Networks

Jae-Gil Lee, *Member*, *IEEE*, Jiawei Han, *Fellow*, *IEEE*, Xiaolei Li, and Hong Cheng

Abstract—Classification has been used for modeling many kinds of data sets, including sets of items, text documents, graphs, and networks. However, there is a lack of study on a new kind of data, *trajectories on road networks*. Modeling such data is useful with the emerging GPS and RFID technologies and is important for effective transportation and traffic planning. In this work, we study methods for classifying trajectories on road networks. By analyzing the behavior of trajectories on road networks, we observe that, in addition to the locations where vehicles have visited, the *order* of these visited locations is crucial for improving classification accuracy. Based on our analysis, we contend that (frequent) sequential patterns are good feature candidates since they preserve this order information. Furthermore, when mining sequential patterns, we propose to confine the length of sequential patterns to ensure high efficiency. Compared with closed sequential patterns, these *partial* (i.e., length-confined) sequential patterns allow us to significantly improve efficiency almost without losing accuracy. In this paper, we present a framework for *frequent pattern-based classification* for trajectories on road networks. Our comparative study over a broad range of classification approaches demonstrates that our method significantly improves accuracy over other methods in some synthetic and real trajectory data.

Index Terms—Trajectory classification, frequent pattern-based classification, road network analysis, sequential patterns.

1 Introduction

With recent improvements in GPS and RFID technologies, a tremendous amount of data about moving objects on road networks is being collected. The tracking of moving objects on road networks is becoming increasingly pervasive. GPS devices embedded in vehicles or other sensors on the streets can track a vehicle as it moves throughout the city traffic grid. A recent proposal from the Oregon Department of Motor Vehicles (DMV) suggests that a tracking system should be mandatory for all newly purchased and newly registered vehicles. Although privacy is a major concern of consumers, one study prepared for the US Department of Transportation says, "Less than 7 percent of the respondents expressed concerns about recording their vehicle's movements." One can expect that such trajectory data, without linking to private information, will become increasingly universal in the near future.

The importance of data analysis over trajectory data is being widely recognized [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]. This paper deals with the problem of classifying vehicles' trajectories on road networks. Trajectory classification is defined as the process of predicting a vehicle's class label based on its trajectory. Trajectory classification has many useful applications, such as city and transportation planning; road construction, design, and maintenance; traffic congestion recognition; law enforcement; and homeland security. In addition, it can be exploited for real-time monitoring [11], e.g., for detecting suspicious movements of vehicles.

Example 1. Intelligent transportation systems have become widespread in recent years. A popular objective of such systems is solving traffic congestion. Trajectory classification can facilitate traffic prediction [12], which is one of the key technologies for solving traffic congestion. Suppose many trip paths (i.e., trajectories of vehicles) are stored in a database. Each of these trips is labeled with its *destination*, such as a district of a town or a city, and they are provided for training. The destination of a new trip is predicted by providing a partial trajectory to the trained classifier. Then, the amount of traffic at specific destinations can be predicted for the near future. If we know that an area will be congested in the near future, traffic can be rerouted.

Example 2. Activity recognition aims at recognizing the actions and goals of a person from a series of observations. The Opportunity Knocks prototype [13] is an example of activity recognition systems. A client transmits GPS signals to a server, and the server makes inferences about the behavior of the client (person) and gives suggestions on what to do next. The system is designed to assist cognitively impaired persons in finding their ways through city traffic. Trajectory classification can facilitate activity recognition since travel paths are closely related to activities. Here, the class label is an *activity*, e.g., commute, shopping, or leisure. Such classification is also useful for marketing since the server can send more appealing advertisements to users according to their current activities.

- *J.-G. Lee is with Department of Knowledge Service Engineering, Korea Advanced Institute of Science and Technology (KAIST), 291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Republic of Korea. E-mail: jaegil@kaist.ac.kr.*
- *J. Han is with the University of Illinois at Urbana-Champaign, Siebel Center for Computer Science, 201 N. Goodwin Avenue, Urbana, IL 61801. E-mail: hanj@cs.uiuc.edu.*
- *X. Li is with Microsoft, Microsoft City Center Plaza, 555 110th Avenue NE, Bellevue, WA 98004. E-mail: xiaoleil@microsoft.com.*
- *H. Cheng is with the Chinese University of Hong Kong, William M.W. Mong Engineering Building, Shatin, N.T., Hong Kong. E-mail: hcheng@se.cuhk.edu.hk.*

Manuscript received 17 June 2008; revised 2 Feb. 2009; accepted 9 Nov. 2009; published online 27 Aug. 2010.
Recommended for acceptance by G. Kollios.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2008-06-0316.
Digital Object Identifier no. 10.1109/TKDE.2010.153.

Feature-based classification has been widely used in the fields of data mining and machine learning [14], [15], [16], [17], [18]. In this paradigm, features are extracted from the data points, and each data point is transformed into a feature vector. Each feature vector represents the existence of features in its corresponding data point. These feature vectors are provided to a classifier such as the support vector machine (SVM) [19] for training and prediction.

One of the most important requirements for effective classification is discovering discriminative features. Thus, the following question naturally arises: "*What are good feature candidates for trajectory classification?*" Recently, the approach using frequent patterns for classification has been successfully adopted for relational data [14], [15], [20], [21], [22], [23], text documents [18], protein sequences [17], and graphs [16]. All of these studies demonstrate the usefulness of frequent patterns in classification. However, to the best of our knowledge, the question on trajectory data is yet to be answered.

To answer this question, we first analyze the behavior of trajectories on road networks. Intuitively, the locations where vehicles have visited are critical features for classification. However, we have found additionally that the *order* of these visited locations is crucial for improving classification accuracy. Based on our analysis, we assert that (frequent) sequential patterns [24] are good feature candidates since they preserve this order information. In experiments, we show that using sequential patterns improves classification accuracy significantly (by 10–15%) over a naive method that uses only individual road segments as features.

Furthermore, we address efficiency issues in trajectory classification because sequential pattern mining from trajectories tends to be time consuming. The length of sequential patterns is confined in the process of sequential pattern mining to enhance classification efficiency. Such *length-confined* sequential patterns are called *partial* sequential patterns. We show that, by using partial sequential patterns instead of *closed* sequential patterns [25], feature generation time is improved significantly (by up to five times) with almost no accuracy loss.

In this paper, we present a framework of *frequent pattern-based classification* for trajectories on road networks. Our framework consists of three steps: 1) feature generation, 2) feature selection, and 3) classification model construction. In the first step, partial sequential patterns are generated. In the second step, highly discriminative ones are selected for effective classification. In the third step, those selected patterns are fed into a classifier. The effectiveness of this framework is verified by comparing classification accuracy and efficiency with other alternatives.

In summary, the contributions of this paper are as follows:

- We analyze the behavior of trajectories on road networks and investigate good feature candidates for trajectory classification.
- We apply frequent pattern-based classification [14] to trajectory classification. More specifically, we present a framework of using *partial* sequential patterns for trajectory classification.
- We conduct a comparative study of a broad range of classification approaches.
- We demonstrate that our frequent pattern-based classification method significantly improves accuracy over other methods and that feature generation can be done efficiently owing to partial sequential patterns.

The rest of the paper is organized as follows: Section 2 presents the problem statement. Section 3 discusses why partial sequential patterns are good feature candidates. Section 4 introduces our trajectory classification method based on partial sequential patterns. Section 5 presents the results of performance evaluation. Section 6 discusses related work. Finally, Section 7 concludes the study.

2 Problem Statement

We develop an accurate and efficient classification method for trajectories on road networks. Especially, we focus on mining discriminative patterns for trajectory classification. Given a set of *trajectories* $\mathcal{D} = \{TR_1, \ldots, TR_{num_{tra}}\}$, with each trajectory associated with a *class label* $c_i \in \mathcal{C} = \{c_1, \ldots, c_{num_{cla}}\}$, our method generates a set of *features*, where the trajectory and feature are defined as follows:

A *road network* is usually modeled as a graph $G\,(V, E)$, where a vertex (i.e., node) of G represents a road junction, an edge a road segment, and an edge weight the distance along the road segment [26]. A *trajectory* is a sequence of edges and is denoted as $TR_i = \langle e_1, e_2, \ldots, e_j, \ldots, e_{len_i}\rangle$ $(1 \leq i \leq num_{tra})$. Here, e_j $(1 \leq j \leq len_i)$ is an edge of G.

Features are extracted from the set of trajectories $\mathcal{D}$. They are categorized into *single features* and *combined features*, which are defined in Definitions 1 and 2, respectively.

Definition 1. *A* single feature *is an edge $e_j \in E$ traversed by at least one trajectory. The set of all single features is denoted as $\mathcal{I}$.*

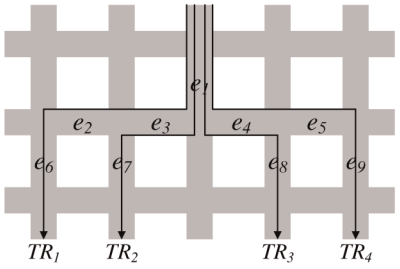
Definition 2. *A* combined feature *is a sequence of single features and is denoted as $\alpha = \langle e_{\alpha_1}, e_{\alpha_2}, \ldots, e_{\alpha_j}, \ldots, e_{\alpha_k}\rangle$ $(k \geq 2)$, where $\forall\, 1 \leq j \leq k$, $e_{\alpha_j} \in \mathcal{I}$.*

To define the frequency of a combined feature, we introduce some necessary notations. Suppose there are two sequences of edges $\alpha = \langle a_1, a_2, \ldots, a_m\rangle$ and $\beta = \langle b_1, b_2, \ldots, b_n\rangle$. *$\beta$ contains α*, denoted as $\alpha \sqsubseteq \beta$, iff $\exists\, i_1, i_2, \ldots, i_m, 1 \leq i_1 < i_2 < \cdots < i_m \leq n$ and $a_1 = b_{i_1} \wedge a_2 = b_{i_2} \wedge \cdots \wedge a_m = b_{i_m}$. $\mathcal{D}_\alpha$ denotes the set of trajectories in $\mathcal{D}$ which contains α, i.e., $\mathcal{D}_\alpha = \{TR | TR \in \mathcal{D} \wedge \alpha \sqsubseteq TR\}$. We now define *frequent* combined features in Definition 3.

Definition 3. *A combined feature α is* frequent *if $\theta \geq \theta_0$, where $\theta = \frac{|\mathcal{D}_\alpha|}{|\mathcal{D}|}$ is the relative support of α, and θ_0 is the minimum support threshold $(0 \leq \theta_0 \leq 1)$. The set of frequent combined features is denoted as $\mathcal{F}$.*

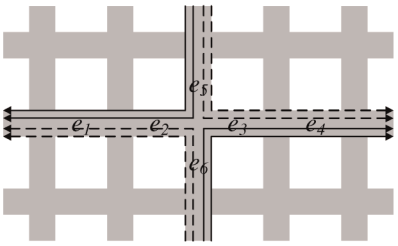
Example 3. Fig. 1 shows a simple road network and four vehicles' trajectories. Road segments are represented by light-shaded thin rectangles, and trajectories by arrow lines. Then, the set of single features is $\mathcal{I} = \{e_1, e_2, \ldots, e_9\}$. If the minimum support threshold θ_0 is 0.5, the set of frequent combined features is $\mathcal{F} = \{\langle e_1, e_3\rangle, \langle e_1, e_4\rangle\}$.

Frequent pattern-based classification takes advantage of frequent combined features. More specifically, our classification method exploits frequent combined features as well as single features. We note that frequent combined features



$\mathcal{I} = \{e_1, e_2, \ldots, e_9\}$, $\mathcal{F} = \{\langle e_1, e_3\rangle, \langle e_1, e_4\rangle\}$ $(\theta_0 = 0.5)$

Fig. 1. An example of single and combined features.



$\rightarrow$: trajectories of c_1, $\dashrightarrow$: trajectories of c_2

Fig. 2. An advantage of using combined features.

are exactly *(frequent) sequential patterns* [24] except for 1-item sets. Frequent combined features do not include 1-item sets since they are already included in single features. Hereafter, the term "sequential pattern" is used to mean a frequent combined feature.

3 Systematic Analysis for Discriminative Patterns

In this section, we discuss why partial sequential patterns are good feature candidates. Our reasoning consists of two steps. First, in Section 3.1, we explain why sequential patterns are useful for trajectory classification. Second, in Section 3.2, we explain why *partial* sequential patterns are more beneficial to trajectory classification than *closed* sequential patterns [25].

3.1 Usefulness of Sequential Patterns

Sequential patterns are useful because 1) each pattern is a sequence of single features, and 2) they are frequent. We explore these two reasons in detail.

3.1.1 Reason I: Sequence of Single Features

A sequential pattern is a frequent sequence of single features, i.e., a frequent combined feature. Thus, sequential patterns preserve the visiting order of road segments (i.e., edges), whereas single features do not. In other words, sequential patterns indicate not only *which road segments are visited*, but also *in which order road segments are visited*; in contrast, single features indicate only which road segments are visited.

This ordering information is indeed crucial for improving the accuracy of trajectory classification since trajectories are spatiotemporal data. Notice that sequential patterns capture both spatial and temporal (i.e., ordering) information, whereas single features capture only spatial information.

Example 4. Fig. 2 shows a bunch of vehicles' trajectories on a road network. Solid lines denote the trajectories of a class c_1, and dashed lines those of a class c_2. The single features e_1–e_6 are useless for distinguishing between the trajectories of c_1 and c_2 since all e_1–e_6 are visited twice by the trajectories of c_1 and also twice by those of c_2. In contrast, combined features are indeed useful for distinguishing between the trajectories of the two classes. $\langle e_5, e_2, e_1\rangle$ and $\langle e_6, e_3, e_4\rangle$ appear only in c_1, and $\langle e_5, e_3, e_4\rangle$ and $\langle e_6, e_2, e_1\rangle$ only in c_2. This simple example intuitively demonstrates the usefulness of combined features.

Sequential patterns [24] used in this paper are different from frequent patterns [27] used by Cheng et al. [14], [15]. The former takes account of the ordering of elements, whereas the latter does not. Obviously, frequent patterns are not adequate for our problem since they cannot capture the ordering information. For example, suppose there are two sets of vehicles driving from Chicago to Champaign and from Champaign to Chicago, respectively. Frequent patterns cannot distinguish between the two sets, whereas sequential patterns can. However, the cost of mining sequential patterns is more expensive than that of mining frequent patterns. Thus, we address efficiency issues in Section 3.2.

3.1.2 Reason II: High Discriminative Power

The discriminative power of a pattern is closely related to its frequency (i.e., support). Cheng et al. [14] have formally investigated this relationship in the context of relational data. Their results demonstrate that patterns of low support have very limited discriminative power due to their limited coverage in data, and patterns of very high support have very limited discriminative power due to their commonness in data.

Fig. 3 shows two types of nondiscriminative patterns, which are illustrated by dark-shaded road segments. Type-I represents the patterns of low support, and Type-II those of very high support. Type-I patterns can be caused by drivers' mistakes (e.g., by taking a wrong path inadvertently) or by the errors of GPS devices. Type-II patterns often occur at bridges or highways since most vehicles may have to pass some bridges or highways to reach a certain place.

Sequential patterns are likely to have high discriminative power since they do not contain Type-I nondiscriminative patterns. By using a proper support threshold, we are able

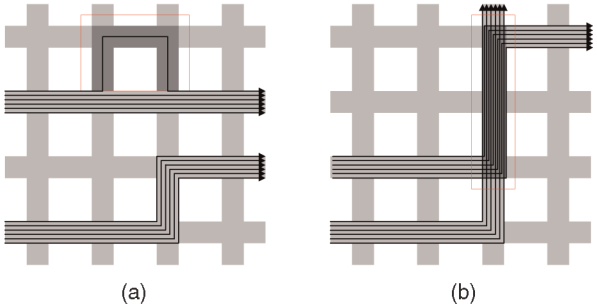


Fig. 3. Two types of nondiscriminative patterns in road networks. (a) Type-I: low support. (b) Type-II: very high support.

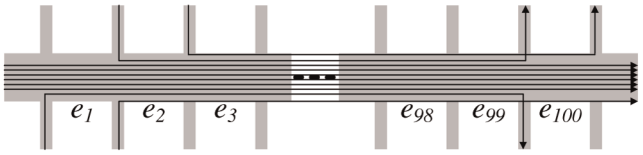


Fig. 4. An example of long sequential patterns.

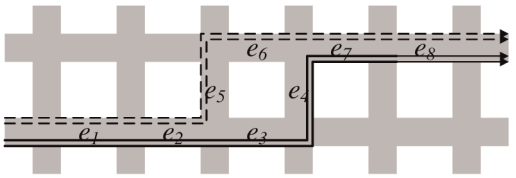


Fig. 5. The discriminative power of partial sequential patterns.

to filter out less frequent sequential patterns. Sequential patterns, however, still contain Type-II nondiscriminative patterns. Type-II patterns will be removed by feature selection.

3.2 Benefits of *Partial* Sequential Patterns

Although sequential patterns are found to be very useful for trajectory classification, mining sequential patterns from trajectories could take a long time. This problem is mainly caused by long sequential patterns since trajectories can be very long. Unfortunately, when mining long sequential patterns or when using a very low support threshold, the performance of sequential pattern mining algorithms often degrades drastically [25].

Example 5. Fig. 4 shows a typical case where long sequential patterns occur. The thick rectangle represents a long popular road or highway, e.g., I-90/94 in Chicago. Thin rectangles represent small roads that join with this long road. A lot of vehicles are running along the long road, and vehicles are continuously merging onto or going out of the long road. As a result, many long sequential patterns exist, e.g., $\langle e_1, e_2, e_3, \ldots, e_{98}, e_{99}, e_{100}\rangle$, $\langle e_2, e_3, \ldots, e_{98}, e_{99}, e_{100}\rangle$, $\langle e_2, e_3, \ldots, e_{98}, e_{99}\rangle$, *etc.*

To ensure high efficiency in sequential pattern mining, we confine the length of sequential patterns. We define *partial* sequential patterns in Definition 4, which are derived from a closed sequential pattern. A sequential pattern α is *closed* if there exists no sequential pattern α' such that $\alpha \sqsubseteq \alpha'$ as well as α and α' have the same support [25]. Notice that closed sequential patterns are chosen as a basis since they are not redundant.

Definition 4. *Suppose $\alpha = \langle e_1, e_2, \ldots, e_n\rangle$ is a closed sequential pattern. The set of k-*partial *sequential patterns ($k \leq n$) of α is defined as $\{\beta | \beta = \langle e_{p_1}, e_{p_2}, \ldots, e_{p_k}\rangle, 1 \leq p_1 < p_2 < \cdots < p_k \leq n\}$. Here, k is called the* length *of partial sequential patterns.*

Example 6. Suppose $\alpha = \langle e_1, e_2, e_3, e_4, e_5\rangle$ is a closed sequential pattern. Then, the set of 4-partial sequential patterns of α is $\{\langle e_1, e_2, e_3, e_4\rangle, \langle e_1, e_2, e_3, e_5\rangle, \langle e_1, e_2, e_4, e_5\rangle, \langle e_1, e_3, e_4, e_5\rangle, \langle e_2, e_3, e_4, e_5\rangle\}$.

We claim that using partial sequential patterns achieves *high efficiency* in feature generation and, at the same time, *high accuracy* almost identical to the accuracy of using closed sequential patterns. Informally, many of the partial sequential patterns are likely to have the same discriminative power as their closed sequential patterns. However, if the length of partial sequential patterns is too short, some of them are likely to be shared by multiple closed sequential patterns of different classes. In this case, partial sequential patterns may not preserve the discriminative power of their closed sequential patterns. Please see Appendix for theoretical analysis on our intuition.

Example 7. Fig. 5 shows four vehicles' trajectories on a road network. Let the minimum support threshold be 0.5. Two closed sequential patterns $\langle e_1, e_2, e_3, e_4, e_7, e_8\rangle$ and $\langle e_1, e_2, e_5, e_6, e_7, e_8\rangle$ are found. All of 5-partial sequential patterns, e.g., $\langle e_1, e_2, e_3, e_4, e_7\rangle$, have the same discriminative power as their closed sequential patterns. This condition holds for some of 2-partial sequential patterns, e.g., $\langle e_3, e_4\rangle$ and $\langle e_5, e_6\rangle$, but not for some of them, e.g., $\langle e_1, e_2\rangle$ and $\langle e_7, e_8\rangle$. Hopefully, we expect that a large proportion of partial sequential patterns will preserve their original discriminative power unless their length is too short.

One might argue that partial sequential patterns of limited discriminative power, e.g., $\langle e_1, e_2\rangle$ and $\langle e_7, e_8\rangle$, could harm classification accuracy. This is partly true, but most of nondiscriminative ones can be removed by feature selection. Thus, accuracy should still be high in most cases with appropriate feature selection. This has been verified by our experiments. Hereafter, we call partial sequential patterns simply as sequential patterns unless we have to distinguish between them.

4 Trajectory Classification Based on Sequential Patterns

In this section, we present the framework of frequent pattern-based classification. This framework consists of three steps: 1) feature generation, 2) feature selection, and 3) classification model construction. We describe these three steps through Sections 4.1, 4.2, and 4.3, respectively. The classification algorithm is summarized in Section 4.4. We finally discuss an extension to use numerical attributes for classification in Section 4.5.

Before proceeding, we summarize the notation to be used throughout the paper in Table 1.

4.1 Feature Generation

We perform sequential pattern mining over the set of trajectories $\mathcal{D}$ for feature generation. In principle, any sequential pattern mining algorithms can be employed for this step. Obviously, we can benefit from the state-of-the-art algorithms.

A very important objective is not to miss any sequential patterns whose discriminative power exceeds a given threshold. We take the *information gain* to measure discriminative power. The information gain threshold is usually

TABLE 1
The Summary of the Notation

	Definition
$\mathcal{I}$	The set of single features
$\mathcal{I}_s$	The result of feature selection over $\mathcal{I}$
$\mathcal{F}(x)$	The set of sequential patterns mined from x
$\mathcal{F}_s(x)$	The result of feature selection over $\mathcal{F}(x)$

given by a user, and a number of strategies on how to set it up have been developed in feature selection methods [28]. The remaining task is to convert the information gain threshold into a minimum support threshold, so that the latter can be provided to sequential pattern mining algorithms. We summarize our strategy for determining the minimum support threshold min_sup in Section 4.1.1.

Another important objective is to guarantee highly efficient feature generation. This objective is accomplished by confining the length of sequential patterns. We present a simple heuristic for determining the maximum length max_len of partial sequential patterns in Section 4.1.2. In fact, this parameter is effective only when the length of a closed sequential pattern x_{closed} is greater than max_len. In this case, max_len-partial sequential patterns are extracted instead of x_{closed}. Otherwise, x_{closed} itself is generated as a feature. Therefore, the length of every sequential pattern is guaranteed to be less than or equal to max_len.

4.1.1 The Minimum Support Threshold

We first derive the theoretical information gain upper bound as done by Cheng et al. [14]. For a pattern represented by a random variable X and classes represented by a random variable C, the information gain is defined as (1). Here, $H(C)$ is the *entropy*, and $H(C|X)$ is the *conditional entropy*. Since $H(C)$ is constant for a given data set, the upper bound IG_{ub} of the information gain is defined as (2). Here, $H_{lb}(C|X)$ means the lower bound of $H(C|X)$

$$IG(C|X) = H(C) - H(C|X), \tag{1}$$
$$IG_{ub}(C|X) = H(C) - H_{lb}(C|X). \tag{2}$$

$H(C|X)$ is defined in (3). Here, we introduce the notation for $H(C|X)$. $P(x)$ means the probability of a pattern x, and $P(\bar{x})$ that of the absence of x. That is, $P(x)$ is equal to the relative support θ of x. $P(c_i)$ means the probability of a class c_i. $P(c_i|x)$ means the probability of c_i if $\exists\, TR \in \mathcal{D}$, $x \sqsubseteq TR$, and $P(c_i|\bar{x})$ that of c_i if $\neg\exists\, TR \in \mathcal{D}$, $x \sqsubseteq TR$. That is, $P(c_i|x)$ indicates what proportion of the trajectories that contain the pattern x belongs to the class c_i

$$\begin{aligned} H(C|X) =& -P(x) \sum_{i=1}^{num_{cla}} P(c_i|x) \log P(c_i|x) \\ & - P(\bar{x}) \sum_{i=1}^{num_{cla}} P(c_i|\bar{x}) \log P(c_i|\bar{x}). \end{aligned} \tag{3}$$

$H(C|X)$ reaches its lower bound when the pattern x occurs in as fewer classes as possible. For ease of exposition, suppose that $\theta \leq \max(P(c_i))$ and that the pattern x exists only in a class c_i. Then, $H_{lb}(C|X)$ is formulated by (4). If $\theta > \max(P(c_i))$, $H_{lb}(C|X)$ can be derived similarly (although more complex)

$$\begin{aligned} & H_{lb}(C|X)_{|P(c_i|x)=1 \wedge P(c_j|x)_{|j\neq i}=0} \\ &= -P(\bar{x}) \left(\frac{P(c_i) - P(x)}{P(\bar{x})} \log \frac{P(c_i) - P(x)}{P(\bar{x})} + \right. \\ & \qquad \left. \sum_{j \neq i} \frac{P(c_j)}{P(\bar{x})} \log \frac{P(c_j)}{P(\bar{x})} \right) \\ &= -(P(c_i) - \theta) \log \frac{P(c_i) - \theta}{1 - \theta} - \sum_{j \neq i} P(c_j) \log \frac{P(c_j)}{1 - \theta}. \end{aligned} \tag{4}$$

Example 8. Suppose there are three classes c_1, c_2, and c_3 with $P(c_1) = 0.4$, $P(c_2) = 0.3$, and $P(c_3) = 0.3$. To make the conditional entropy *minimum*, a pattern x should appear in only one class. Let's say it is c_1. Then, (4) is formulated as below:

$$H_{lb}(C|X) = -(0.4 - \theta) \log \frac{0.4 - \theta}{1 - \theta} - 2 \cdot 0.3 \log \frac{0.3}{1 - \theta}.$$

min_sup is derived from an information gain threshold IG_0 [14]. We first compute the theoretical information gain upper bound as a function $IG_{ub}(\theta)$[1] of the support θ. This function involves only the class label distribution. Next, we choose IG_0 and find $\theta^* = \arg\max_\theta(IG_{ub}(\theta) \leq IG_0)$. Finally, we mine sequential patterns using $min\_sup = \theta^*$. Combined features with the support $\theta < \theta^*$ can be removed early on because $IG(\theta) \leq IG_{ub}(\theta) < IG_{ub}(\theta^*) \leq IG_0$. In this way, sequential patterns are generated efficiently without missing any feature candidates with respect to IG_0.

4.1.2 The Length of Partial Sequential Patterns

We take account of *classification accuracy* and *feature generation time* in determining max_len of partial sequential patterns. The former might get worse as max_len decreases. In contrast, the latter is improved as max_len decreases since sequential pattern mining can be completed earlier. Thus, the optimal max_len is the smallest one that has almost no accuracy loss. Finding the optimal max_len is complicated. We do not consider it here, leaving it as the topic of a future paper. Fortunately, classification accuracy is shown to be rather insensitive to max_len.

Instead, we use a simple heuristic that increases max_len as long as sequential pattern mining is completed in a reasonable time. This heuristic prevents possible loss of classification accuracy. Our experience indicates that max_len is usually determined to be around 5. Please see Appendix for theoretical analysis.

4.2 Feature Selection

The objective of the feature selection step is to select highly discriminative sequential patterns. We primarily *filter out Type-II nondiscriminative patterns* shown in Fig. 3b. They have very limited discriminative power despite very high support. Notice that we perform feature selection over only the set of sequential patterns $\mathcal{F}(\mathcal{I})$. Feature selection over the set of single features $\mathcal{I}$ can improve classification

1. $IG_{ub}(\theta)$ stands for the upper bound of the information gain when the argument is assigned to the value of θ.

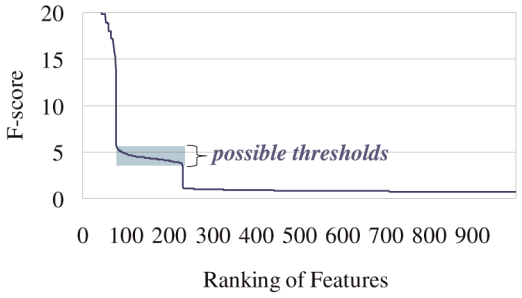


Fig. 6. An example of the F-score distribution.

accuracy to some extent, but we observe that the improvement is marginal in many cases. In principle, any feature selection algorithms can be employed for this step.

We adopt the feature selection method using the *F-score* [29].[2] The F-score is a variant of the Fisher score. The F-score[3] of the ith feature is

$$F-score(i) = \frac{(\bar{a}_i^{(+)} - \bar{a}_i)^2 - (\bar{a}_i^{(-)} - \bar{a}_i)^2}{\frac{1}{n^+-1}\sum_{k=1}^{n^+}(a_{k,i}^{(+)} - \bar{a}_i^{(+)})^2 - \frac{1}{n^--1}\sum_{k=1}^{n^-}(a_{k,i}^{(-)} - \bar{a}_i^{(-)})^2}, \quad (5)$$

where n^+ (or n^-) is the number of positive (or negative) instances; $\bar{a}_i$ is the average of the ith feature of all instances; $\bar{a}_i^{(+)}$ (or $\bar{a}_i^{(-)}$) is the average of the ith feature of positive (or negative) instances; and $a_{k,i}^{(+)}$ (or $a_{k,i}^{(-)}$) is the ith feature of the kth positive (or negative) instance.

Equation (5) is intuitively described as follows [29]: "The numerator indicates the discrimination between the positive and negative sets, and the denominator indicates the one within each of the two sets." The larger the F-score is, the more likely this feature is discriminative. Hence, we use this score as a feature selection criterion.

Then, the remaining task is to decide how many sequential patterns should be selected for effective classification. We adopt the strategy provided with the F-score method. First, we calculate the F-scores of sequential patterns, and then, pick some thresholds. As in Fig. 6, possible thresholds are identified by plotting F-scores on a graph and picking a few values from the interval where the F-score begins to be stable. Next, for each threshold, we filter out sequential patterns whose F-score is below the threshold, and then, conduct *k-fold cross validation* [31] over the filtered training data. Finally, we choose the threshold with the lowest average validation error. The number of sequential patterns selected is automatically determined by this F-score threshold.

4.3 Classification Model Construction

We use the union of single features and sequential patterns selected, i.e., $\mathcal{I} \cup \mathcal{F}_s(\mathcal{I})$, as features. Thus, we map each trajectory into a feature vector in the feature space of $\mathcal{I} \cup \mathcal{F}_s(\mathcal{I})$. Each entry of a feature vector corresponds to a feature, either a single feature or a sequential pattern. The ith entry of a feature vector is equal to the frequency that the ith feature occurs in the trajectory. Notice that this mapping should be performed for both the training set and the test set although sequential patterns are discovered by mining only the training set.

In our study, we build a classification model using the support vector machine (SVM) [19]. This design decision stems from two characteristics of the feature vectors generated. First, they are *high dimensional* since many single features (i.e., road segments) may exist even in a small road network (typically, more than 1,000). Furthermore, quite a large number of sequential patterns may be discovered from these single features (typically, more than 10,000). Second, they are *sparse* since each trajectory has only a few of these features. The SVM is well suited for such high dimensional and sparse data [31].

Algorithm **Trajectory Classification**

INPUT: A set of trajectories $\mathcal{D} = \{TR_1, \ldots, TR_{num_{tra}}\}$, with each trajectory associated with a class label c_i
OUTPUT: A classification model
ALGORITHM:
/* 4.1: FEATURE GENERATION */
01: Determine the minimum support threshold min_sup;
02: Determine the length of sequential patterns max_len;
03: Perform sequential pattern mining over $\mathcal{D}$;[4]
04: Obtain the set of sequential patterns $\mathcal{F}(\mathcal{I})$;
/* 4.2: FEATURE SELECTION */
05: Calculate the F-scores of sequential patterns;
06: Determine the number of sequential patterns selected;
07: Obtain the result of feature selection $\mathcal{F}_s(\mathcal{I})$;
/* 4.3: CLASSIFICATION MODEL CONSTRUCTION */
08: Translate each trajectory into a feature vector in the feature space of $\mathcal{I} \cup \mathcal{F}_s(\mathcal{I})$;
09: Feed the set of feature vectors into a classifier;

Fig. 7. The frequent pattern-based classification algorithm.

4.4 Algorithm

Fig. 7[4] summarizes the frequent pattern-based classification algorithm. This algorithm is self-explanatory and goes through three steps as discussed above.

4.5 Discussion

Our framework can be easily extended to use numerical attributes for classification. A set of attributes can be attached to each edge of a trajectory. Examples of such attributes include the average speed, top speed, elapsed time, day, time-of-day, *etc.* In this case, different single features are generated from one edge depending on the values of the attributes.

2. We expect that other feature selection methods will lead to similar results. In fact, SVM-Infoprop [30] using the weight vector of the SVM has shown to generate quite similar results.

3. We show the definition for binary classification as in the original literature [29]. This definition can be easily extended to accommodate multiclass classification.

4. Some may consider to use a short sliding window or gap constraints to confine the examination of sequential patterns with small gaps. However, there may exist some long bridges or highway segments, and the distinct combined patterns can only be observed by combining the segments before entering a bridge or after getting off the bridge; thus, a small sliding window or gap constraints may not lead to finding effective patterns. Hence, we do not put sliding window ideas or gap constraints into our design.

TABLE 2
Five Classification Approaches Compared

Approach	Features Used
Single_All	$\mathcal{I}$
Single_DS	$\mathcal{I}_s$
Seq_All	$\mathcal{I} \cup \mathcal{F}(\mathcal{I})$
Seq_PreDS	$\mathcal{I}_s \cup \mathcal{F}(\mathcal{I}_s)$
Seq_DS	$\mathcal{I} \cup \mathcal{F}_s(\mathcal{I})$

Suppose the time-of-day is an important factor for a specific classification application. Then, if a vehicle visits an edge e_i at 9 a.m. and 6 p.m., these two visits should be represented by separate single features, e.g., $(e_i, morning)$ and $(e_i, evening)$.

An interesting issue is to decide a proper resolution for an attribute. In the above example, 9 a.m. can be represented by using various resolutions ranging from a high level to a low level, e.g., *morning* or *8–10 a.m.* A proper resolution needs to be selected based on its discriminative power. Since there are more mature studies on this issue in feature generalization methods [8], we can borrow their strategies and split the attribute domain using the selected resolution.

5 Performance Evaluation

In this section, we evaluate the performance of the frequent pattern-based classification method. We describe the experimental data and environment in Section 5.1. We first report the results for synthetic data in Sections 5.2 and 5.3. The accuracy and efficiency results are provided in each section. We then report the results for real data in Section 5.4.

5.1 Experimental Setting

All experiments are conducted on a Pentium 4 3.0 GHz PC with 1 GB of main memory, running on Windows XP. Besides, LIBSVM [32] with a linear kernel is used to build a classifier.

5.1.1 Classification Approaches

We compare five classification approaches summarized in Table 2. They are categorized depending on the features used. *Single_All* uses all single features, and *Single_DS* selected single features. *Seq_All* uses all single features as well as all sequential patterns, and *Seq_DS* all single features as well as selected sequential patterns. *Seq_PreDS* selects single features, and then, finds sequential patterns from these selected single features. Consequently, the sequential patterns used in *Seq_PreDS* are quite different from those in *Seq_DS*. Among these approaches, *Seq_DS* is what we propose. The feature selection method explained in Section 4.2 is applied to both single features and sequential patterns.

5.1.2 Data Generation

For synthetic data, we use the network-based data generator[5] developed by Brinkhoff [33]. It generates synthetic trajectories on a real-world road network. The synthetic trajectories are quite close to real traffic since they are the fastest routes found by considering the maximum speed and capacity of a road, other moving objects on the road, and other external factors. The data generator, as it is, generates essentially random traffic: starting and ending points are randomly chosen within the network. In order to simulate more realistic real-world data (i.e., more skewed than pure randomness), we modify the data generator in two ways.

First, the starting (or ending) points of trajectories are located close to each other if the trajectories belong to the same class. For each class, six small (five-block-sized) regions are randomly picked, and then, the starting points and the ending points are chosen only from the nodes within these regions. N_{region} denotes the number of such regions. In this way, we make the trajectories of the same class share some subroutes.

Second, most trajectories are forced to pass by a small number of "hot edges." These hot edges are visited in a given order for certain classes, but in a totally random order for other classes. We believe that this data generation is reasonable in a real environment since highways or bridges can become hot edges. Moreover, many edges around popular buildings can also be hot edges, and it is rather common to visit these popular buildings in a fixed order (e.g., a hospital $\rightarrow$ a drug store). N_{hot_edge} denotes the *distinct* number of such hot edges, and N_{hot_class} the number of classes where the visiting order of hot edges is specified. In this way, we control the impact of sequential patterns on trajectory classification.

Fig. 8. Snapshots of 1,000 trajectories generated for two different classes.

Fig. 8 shows the snapshots of 1,000 trajectories generated for two different classes. The road network of City of Stockton in San Joaquin County, which contains 18,496 nodes and 24,123 edges, is used. Half of trajectories are randomly chosen for the training set, and the other half are used for the test set.

10 basic data sets D1–D10 are generated using the following parameter values: for D1–D5, *the number of classes* = 5, *the number of trajectories per class* = 1,000, $N_{region} = 6$, $N_{hot\_edge} = 5$, and $N_{hot\_class} = 1$; for D6–D10, *the number of classes* = 10, *the number of trajectories per class* = 1,000, $N_{region} = 6$, $N_{hot\_edge} = 5$, and $N_{hot\_class} = 2$.

5.1.3 The Minimum Support Threshold

The minimum support threshold is derived from the information gain threshold IG_0 which is given empirically. A good value of IG_0 ranges around 0.2 for our data sets according to our experience. For D1–D5, IG_0 is set to be 0.25, and the function $IG_{ub}(\theta)$ is formulated by (6). The function is plotted as Fig. 9, so the minimum support threshold is determined to be 0.1

$$IG_{ub}(\theta) = -5 \cdot 0.2 \log 0.2 - \left\{ -(0.2-\theta)\log\frac{0.2-\theta}{1-\theta} - 4 \cdot 0.2 \log\frac{0.2}{1-\theta} \right\}. \quad (6)$$

5. http://www.fhoow.de/institute/iapg/personen/brinkhoff/generator/.

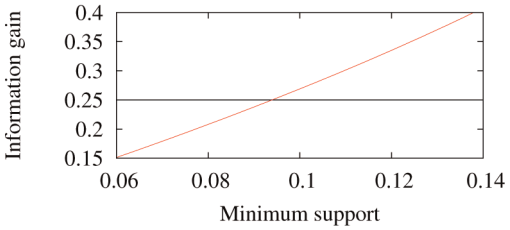


Fig. 9. The plot of $IG_{ub}(\theta)$ for D1–D5.

By using the same strategy, the minimum support threshold for D6–D10 is set to be 0.06, and that for the real data set in Section 5.4 to be 0.2.

5.1.4 Feature Generation and Selection

We use the CloSpan [25] algorithm to find sequential patterns. CloSpan has been regarded as the state-of-the-art algorithm for closed sequential pattern mining. The implementation of CloSpan is slightly modified so as to generate *partial* sequential patterns. At the time when CloSpan tries to extend sequences, the length of a candidate sequence is additionally checked whether it reaches max_length.

Table 3 shows the number of features (i.e., single features and sequential patterns) obtained through feature generation and feature selection. We set the maximum length of sequential patterns to be 5.

5.2 Accuracy Results

5.2.1 Overall Results

Table 4 shows the overall results for accuracy. These results are summarized as follows:

- *Seq_DS* performs the best as discussed in Section 3.1. *Seq_DS* improves accuracy by up to 15 percent over *Single_All*. This is exactly the advantage of using sequential patterns, which we propose in this paper.
- *Single_All* shows low accuracy since single features cannot capture the order of road segments that vehicles have visited.
- *Single_DS* outperforms *Single_All* if many of single features are not discriminative. In contrast, *Single_DS* could decrease accuracy if many of single features are highly discriminative.

TABLE 3
The Number of Features Used

	Single_All	*Single_DS*	*Seq_All*	*Seq_PreDS*	*Seq_DS*
D1	14077	7038	27157	8968	14179
D2	13701	6850	25142	7010	13723
D3	13267	6633	25114	7641	13313
D4	14905	7452	26324	7655	14994
D5	13081	6540	37304	12380	13175
D6	20272	10136	27954	10329	20512
D7	20328	10164	28010	10420	20388
D8	20934	10467	28636	10887	21054
D9	21189	10594	29412	11991	21317
D10	20789	10394	38484	13616	20927

TABLE 4
Classification Accuracy (Percent)

	Single_All	*Single_DS*	*Seq_All*	*Seq_PreDS*	*Seq_DS*
D1	84.88	84.76	77.76	82.32	**94.72**
D2	82.72	83.08	84.84	82.92	**95.68**
D3	86.68	92.40	76.84	89.36	**93.24**
D4	78.04	76.20	78.44	76.44	**89.60**
D5	68.60	68.60	75.64	67.88	**84.04**
D6	78.18	78.40	73.10	77.88	**91.34**
D7	80.56	82.16	77.84	81.88	**91.26**
D8	80.00	81.02	70.26	80.04	**88.34**
D9	70.04	69.68	69.08	67.90	**83.18**
D10	73.38	74.98	68.84	74.86	**86.96**
AVG	78.31	79.13	75.26	78.15	**89.84**

- *Seq_All* shows low accuracy since not all sequential patterns are discriminative. That is, sequential patterns still contain Type-II nondiscriminative patterns in Fig. 3b.
- *Seq_PreDS* performs worse than *Seq_DS* does because in *Seq_PreDS*, some single features are removed too early. As shown in Example 4, sequences of single features can be really useful for classification although these single features themselves are not. *Seq_PreDS* could miss such sequences.

5.2.2 Varying the Length of Sequential Patterns

Fig. 10 shows the classification accuracy of *Seq_DS* as the length of sequential patterns, denoted as max_len, is varied for the data set D3. We observe that we can obtain the near-optimal accuracy when $max\_len \geq 3$. Here, the accuracy loss is as small as 1 percent. As discussed in Section 3.2, many partial sequential patterns are likely to have the same discriminative power as their closed sequential patterns. However, if length is too short (e.g., $max\_len = 2$), some partial sequential patterns are likely to be shared by multiple closed sequential patterns of different classes. This sharing obviously decreases classification accuracy.

5.2.3 Varying the Number of Selected Features

Fig. 11 shows the classification accuracy of *Seq_DS* as the number of selected features is varied for the data set D9. Classification accuracy increases as more sequential patterns with high discriminative power are included. After classification accuracy reaches the maximum, it begins to

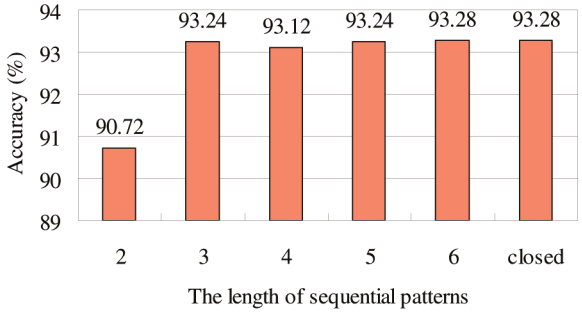


Fig. 10. The effects of the length of sequential patterns on accuracy.

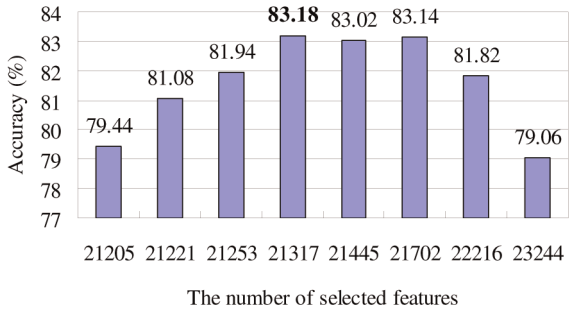


Fig. 11. The effects of the number of selected features on accuracy.

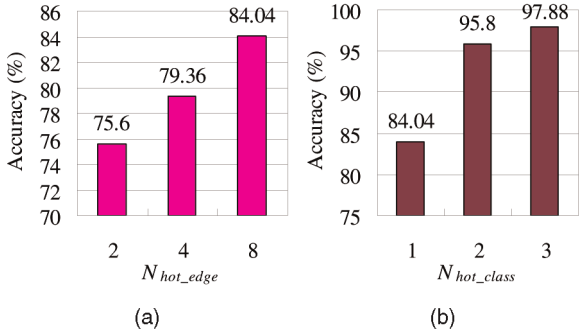


Fig. 12. The effects of N_{hot_edge} and N_{hot_class} on accuracy. (a) N_{hot_edge}. (b) N_{hot_class}.

decrease since the remaining sequential patterns have limited discriminative power. The optimal number of selected features, i.e., 21,317, coincides with the one chosen by our heuristic. The heuristic chooses the one with the lowest average validation error of k-fold cross validation. This result indeed verifies our analysis that not all sequential patterns have high discriminative power.

5.2.4 Varying the Parameters N_{hot_edge} and N_{hot_class}

Fig. 12 shows the classification accuracy of *Seq_DS* as the two parameters N_{hot_edge} and N_{hot_class} are varied for the data set D5. We set N_{hot_class} to be 1 in Fig. 12a and N_{hot_edge} to be 5 in Fig. 12b. We can easily expect that classification accuracy will increase as these parameters get larger. When N_{hot_edge} gets larger, sequential patterns of hot edges are more likely to appear only in the classes where a visiting order has been specified. Hence, those sequential patterns become more discriminative. On the other hand, when N_{hot_class} gets larger, more sequential patterns of hot edges are generated.

5.3 Efficiency Results

5.3.1 Training Time

Table 5 shows the training time of the five approaches. Training time is heavily dependent on the number of features used. Thus, *Single_DS* is the best in terms of training time. We note that the training time of *Seq_DS* is comparable to that of *Single_All*. That is, training time is even decreased or increased only by less than 10 percent in most cases. The reason is that not many sequential patterns are necessary to increase classification accuracy.

TABLE 5
Training Time (Seconds)

	Single_All	*Single_DS*	*Seq_All*	*Seq_PreDS*	*Seq_DS*
D1	3.20	2.30	122.30	4.66	3.13
D2	2.92	2.06	122.90	2.24	2.75
D3	2.48	1.89	113.89	3.04	2.42
D4	2.96	2.30	143.97	2.56	3.85
D5	2.36	1.92	136.17	8.14	2.50
D6	10.78	7.82	425.10	8.33	11.76
D7	9.51	6.90	416.93	7.68	8.80
D8	10.39	7.77	419.24	9.47	9.94
D9	9.40	7.17	419.32	11.84	9.14
D10	9.78	7.03	463.85	18.94	9.48

TABLE 6
Prediction Time (Seconds)

	Single_All	*Single_DS*	*Seq_All*	*Seq_PreDS*	*Seq_DS*
D1	2.93	2.30	127.88	4.10	3.49
D2	2.75	2.00	126.90	2.25	2.88
D3	2.29	1.80	122.38	2.83	2.55
D4	3.20	2.32	136.75	2.62	4.22
D5	2.69	2.06	151.33	9.05	3.02
D6	13.00	9.83	435.74	10.54	16.39
D7	11.08	7.80	428.07	8.99	11.94
D8	12.84	9.47	439.02	11.77	14.24
D9	12.19	9.06	441.13	14.37	13.59
D10	11.64	8.63	481.02	23.09	13.19

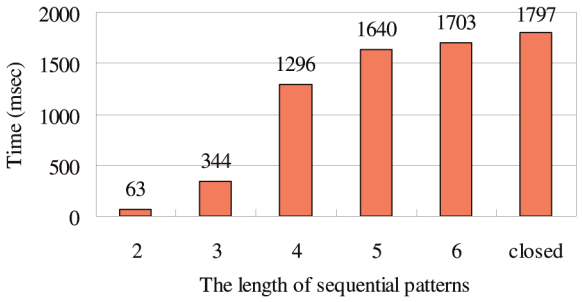


Fig. 13. Feature generation time depending on the length of sequential patterns.

5.3.2 Prediction Time

Table 6 shows the prediction time of the five approaches. The trend in Table 6 is the same as that in Table 5. Again, the prediction time of *Seq_DS* is comparable to that of *Single_All*.

5.3.3 Feature Generation Time

Fig. 13 shows feature generation time as the length of sequential patterns is varied for the data set D3. We do not include the time for reading an input file since this time is the same regardless of the length of sequential patterns. Feature generation time for 3-partial sequential patterns is shown to be as small as 20 percent of that for closed sequential patterns. Nevertheless, accuracy is still high as shown in Fig. 10. This result indeed indicates the advantage of partial sequential patterns. If we use a larger data set, feature generation for

TABLE 7
The Number of Features Used in R1 and R2

	Single_All	*Single_DS*	*Seq_All*	*Seq_PreDS*	*Seq_DS*
R1	9745	4872	39757	15440	10682
R2	12913	6456	30155	18008	13182
R3	16961	8480	32837	24356	17235

closed sequential patterns might not even terminate in a reasonable time. In such cases, the advantage of partial sequential patterns becomes more prominent.

5.4 Results for Real Data

Real-world data are used to test the effectiveness of our classification method. 24 days of taxi data in the San Francisco area were collected during the month of July in 2006. In all, there were over 800,000 separate trips, 33 million road-segment traversals, and 100,000 distinct road segments. Each trajectory represents a trip from when a driver picks up passengers to when the driver drops them off. The name of the taxi company cannot be revealed for confidentiality reasons.

Since there is originally no labeling on the data, we select part of the data and prepare *two*-class data sets as follows: The most frequently visited road segments are first identified, and two of them are chosen for each data set. The trajectories of one class include the two road segments in a given order, but those of the other class in the opposite order. The data set R1 is generated using Bayshore Freeway and Market Street,[6] and the data set R2 using Interstate 280 and US Route 101. The number of trajectories in R1 and R2 are 1,123 and 2,455, respectively. In addition to R1 and R2, a *four*-class data set R3 is generated by just combining R1 and R2. Two classes each from R1 and R2 constitute four classes.

Table 7 shows the number of features used in the three data sets R1, R2, and R3. The difference between *Single_All* and *Seq_DS* is typically small, which means that only a small number of highly discriminative sequential patterns are necessary to increase classification accuracy. The number of sequential patterns in R3 is smaller than that in R1 because the same relative minimum support is used for R1, R2, and R3.

Figs. 14a and 14b show the classification accuracy of the five approaches for the two data sets R1 and R2. *Seq_DS* performs the best also in real data sets. *Seq_DS* improves accuracy by about 4 percent over *Single_All*. This result is very natural. In the data set R1, one class mainly represents traffic *into* downtown San Francisco, and the other class traffic *out of* downtown San Francisco. Both classes contain Market Street, its nearby road segments, and some segments of Bayshore Freeway, but the visiting sequences of these road segments are different for the two classes. Thus, *Single_All* may not distinguish between the two classes, whereas *Seq_DS* can.

In Fig. 14c, the result for the four-class data set R3 is presented. This result is shown to be very similar to Figs. 14a and 14b, but the overall accuracy is slightly reduced. It is common that classification accuracy decreases as the number of classes increases. Our method *Seq_DS* still outperforms other approaches.

6. Market Street is a major street and important thoroughfare in downtown San Francisco.

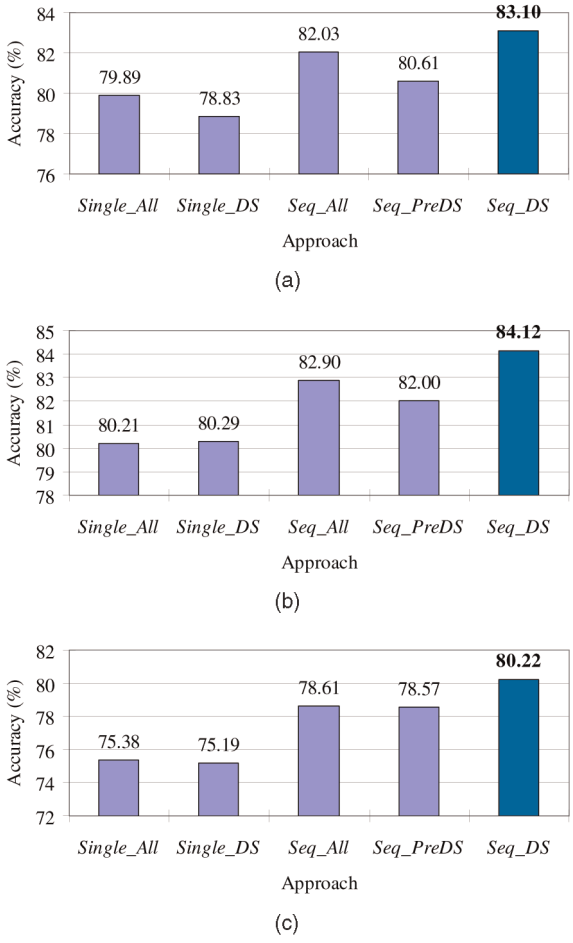


Fig. 14. Classification accuracy for real data sets. (a) R1: the Bayshore-Market data set. (b) R2: the I280-US101 data set. (c) R3: the combined four-class data set.

6 Related Work

6.1 Classification

The concept of frequent pattern-based classification has been introduced in associative classification. Representative methods include CBA [22], CMAR [21], CPAR [23], and RCBT [20]. In these methods, association rules are generated and analyzed for use in classification. The methods search for strong associations between frequent patterns (conjunctions of attribute-value pairs) and class labels. Classification is done by evaluating a set of rules in the form of "$p_1 \wedge p_2 \wedge \cdots \wedge p_l \Rightarrow A_{class} = C$." In many studies, associative classification has been found to be more accurate than some traditional classification methods such as C4.5.

Cheng et al. [14] have proposed a frequent pattern-based classification method for relational data. Efficiency issues have also been addressed in subsequent work [15]. This method uses frequent patterns as combined features, whereas ours uses sequential patterns. According to our categorization using the type of features in Table 2, this method falls into $\mathcal{I} \cup \mathcal{F}_s(\mathcal{I})$. The authors have provided

in-depth analysis on why frequent patterns provide a good solution for classification and investigated the relationship between discriminative power and pattern frequency. By using this relationship, the authors have developed a formal strategy for determining the minimum support threshold. Our method is based on this work.

A number of frequent pattern-based classification methods have been developed in different domains. Lodhi et al. [18] have proposed a classification method for text documents, Leslie et al. [17] for protein sequences, and Deshpande et al. [16] for graphs, where phrases, substrings, and subgraphs are used as features. According to our categorization, these methods fall into $\mathcal{F}(\mathcal{I})$ or $\mathcal{F}_s(\mathcal{I})$. In all these studies, frequent patterns are generated, and the data are mapped to a higher dimensional feature space. Data which are not separable in the original space become linearly separable in the mapped space. These methods, however, are not tailored to trajectory classification as opposed to ours.

Fraile and Maybank [11] have applied a hidden Markov model (HMM) [34] to classifying vehicles' trajectories. This method is similar to ones successfully applied to speech recognition. The measurement sequence is first divided into overlapping segments. In each segment, the trajectory of a vehicle is approximated by a smooth function, and then, assigned to one of the four categories: ahead, left, right, or stop. In this way, the list of segments is reduced to a string of symbols drawn from the set $\{a, l, r, s\}$. The string of symbols is classified using the HMM. That is, an HMM for the motion of the vehicle is constructed, and the Viterbi [34] algorithm is used to find the sequence of internal states for which the observed behavior of the vehicle has the highest probability. This method, however, cannot exploit a sequence of visited locations because, in the Markov chain, the probability of each state x_i depends only on the value of x_{i-1}. In contrast, discriminative patterns could be many states apart.

There are more methods developed for time series and general trajectories. The method in [35] basically belongs to *Single_DS* which is one of our alternatives. Thus, it is already covered by the current experiments. The method in [36] is not appropriate for the trajectories on road networks since it performs classification using the distance between *entire* time series. There is no well-defined distance measure between two entire trajectories *on road networks*. Moreover, discriminative features are likely to exist on some *parts* of the trajectories. The method in [37] is not appropriate either since it assumes that the trajectories are generated by continuous functions. The sequences of road segments are hard to be modeled by continuous functions.

6.2 Trajectory Pattern Mining

In this section, we briefly review recent studies on trajectory pattern mining. Please notice that these studies are on mining different trajectory patterns, but not on classification based on trajectory patterns. Moreover, our framework is general enough to adopt new kinds of trajectory patterns with proper extensions.

Giannotti et al. [2] have developed an algorithm of mining trajectory patterns. A *T-pattern* is $T = (S, A)$, where $S = \langle s_0, \ldots, s_n \rangle$ is a sequence of $n+1$ locations, and $A = \langle \alpha_1, \ldots, \alpha_n \rangle$ is a sequence of transition times between consecutive locations. When generating T-patterns, similar locations should be considered as being the same since the exactly same location usually never occurs. To handle spatial similarity, *regions of interest* (i.e., popular regions) are discovered based on density. Then, T-patterns are represented using only these regions.

Gudmundsson and Kreveld [4] have proposed methods of finding longest duration flocks in trajectory data. A *flock* in a time interval I, where the duration of I is at least k, consists of at least m entities such that for every point in time within I, there is a disk of radius r that contains all the m entities. Compared with T-patterns, intermediate paths between consecutive regions and absolute times are important in flocks, whereas they are not in T-patterns. Flocks could be used as features for trajectory classification. However, there is no need to find the *longest* duration flocks for classification.

T-patterns and flocks are designed to support *general* trajectories, i.e., sequences of (x, y, t), but are not tailored to trajectories on road networks. To the best of our knowledge, most data mining algorithms for trajectories on road networks assume the environment where raw trajectories are already transformed to sequences of road segments by map matching algorithms. This environment is natural since the sequences of road segments are much more intuitive and concise than those of GPS points. Furthermore, the output of map matching is very reliable in recent days. Therefore, there is no need to use the algorithms for general trajectories in our paper.

Mamoulis et al. [9] and Cao et al. [1] have proposed methods of discovering periodic patterns in spatiotemporal sequences. A *periodic pattern* is defined as a T-length sequence of the form $r_0 r_1 \ldots r_{T-1}$, where r_i is a spatial region or $*$ (the whole spatial universe). Although periodic patterns could be used as features, the patterns found in this work are within the trajectory of a *single* moving object. In order to be useful for classification, the patterns should be within the trajectories of many moving objects in the same class. This work is not tailored to trajectories on road networks either.

Zheng et al. [10] have proposed a method of mining interesting locations and travel sequences from GPS trajectories. The main idea is to exploit users' travel experiences. A user's visit to a location is regraded as a direct link from the user to the location. Then, an HITS-based model is used to infer the user's travel experience and the interest of the location. However, high interestingness does not necessarily mean high discriminative power since interestingness increases as the location is visited more often. Recall that Type-II patterns in Fig. 3b have limited discriminative power. This work is not tailored to trajectories on road networks either.

Gidófalvi and Pedersen [3] have proposed methods of mining long, sharable patterns (LSP) from trajectories on road networks. Since their main target is a ride-sharing application, long patterns are definitely preferable for their purposes. However, long patterns are not very critical to classification accuracy as proven in our paper. The LSP mining algorithm tries to mine frequent patterns (not sequential patterns) mainly due to performance issues since patterns in trajectories could be extremely long. Thus, LSPs are not suitable for classification purposes as discussed in Section 3.1.1.

TABLE 8
The Summary of the Notation

Symbol	Definition
θ	$P(x=1)$, *i.e.*, the relative support
p	$P(c=1)$
q	$P(c=1\|x=1)$
n	the number of *closed* sequential patterns
l	the length of *closed* sequential patterns
k	the length of *partial* sequential patterns

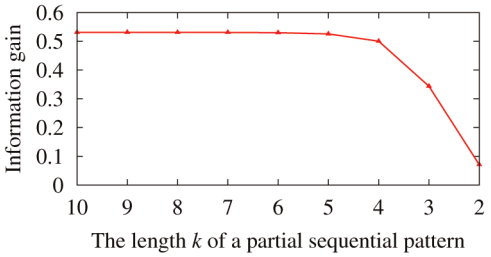


Fig. 15. Estimation of the information gain as k varies.

7 Conclusions

In this paper, we have presented a framework of frequent pattern-based classification for trajectories on road networks. We have performed systematic analysis about the behavior of trajectories on road networks. Our analysis leads to the conclusion that partial sequential patterns are good feature candidates.

We have conducted extensive experiments by comparing classification accuracy and efficiency of five classification approaches. Experimental results show that *Seq_DS* improves classification accuracy by up to 15 percent over *Single_All*. At the same time, training time and prediction time are shown to increase only by less than 10 percent, and feature generation time is shown to decrease significantly with classification accuracy losing by as small as 1 percent. Also, in real data sets, *Seq_DS* achieves the highest classification accuracy among the five approaches. These results indeed demonstrate the effectiveness of trajectory classification using partial sequential patterns.

Overall, we believe that we have provided a practical framework for trajectory classification which can be used in a real environment. Also, we believe that the methodology developed here tells us that using sophisticated patterns can be quite useful in classification. Further study on efficient and effective methods for pattern-based classification with more sophisticated patterns is an interesting direction for future research.

Appendix

We theoretically show that a large proportion of partial sequential patterns will preserve their original discriminative power unless their length is too short. For ease of exposition, binary classification is used for our theoretical analysis. X denotes a random variable for a pattern, and C that for a class. Suppose $X \in \{0,1\}$ and $C \in \{0,1\}$ due to binary classification. Table 8 summarizes the notation for this analysis.

The conditional entropy of a closed sequential pattern is formulated by (7). The information gain can be easily obtained by (1) and is employed as the measure of discriminative power

$$\begin{aligned} H(C|X) = &-\theta q \log q - \theta(1-q)\log(1-q) \\ &+ (\theta q - p)\log\frac{p-\theta q}{1-\theta} \\ &+ (\theta(1-q)-(1-p))\log\frac{(1-p)-\theta(1-q)}{1-\theta}. \end{aligned} \tag{7}$$

If a k-partial sequential pattern is used instead of a closed sequential pattern, q needs to be changed accordingly as (8). One of k-permutations from a closed sequential pattern in the other class may match the k-partial sequential pattern, which makes q decrease by that probability. Equation (8) is a simplified formulation, but it is enough for a ballpark analysis. We assume $\theta_k \cong \theta_{closed}$ though θ_k will actually get a little larger. p is the same regardless of the length of a partial sequential pattern

$$q_k = q_{closed} - \frac{n(1-p)}{{}_lP_k}. \tag{8}$$

We plot the information gain of a k-partial sequential pattern in Fig. 15 with k varied from 10 down to 2. Other variables are set reasonably as follows: $\theta = 0.5$, $p = 0.5$, $q = 0.9$, $n = 100$, and $l = 10$. The information gain is kept high when $k \geq 4$, drops rapidly when $k = 3$, and becomes very low when $k = 2$. Thus, if we set k to be 2, classification accuracy would not be as good. This analysis coincides very well with Fig. 10 in Section 5.2.2.

It turns out that this analysis is very helpful for determining max_len. We can draw a graph just like Fig. 15 by using the information of a given data set. Then, it is required to find the length where the information gain decreases significantly, e.g., $k = 3$ in Fig. 15. $\max\_len$ can be the smallest value larger than the length. Our experience indicates that $max\_len = 5$ is a good configuration, in general.

Acknowledgments

The work was supported in part by the Korea Research Foundation grant, funded by the Korean Government (MOEHRD), KRF-2006-214-D00129, and in part by the US National Science Foundation (NSF) grants IIS-09-05215, CNS-0931975, and BDI-07-Movebank. Any opinions, findings, and conclusions expressed here are those of the authors and do not necessarily reflect the views of the funding agencies. This work was done while the first, third, and fourth authors were at the University of Illinois.

References

[1] H. Cao, N. Mamoulis, and D.W. Cheung, "Discovery of Periodic Patterns in Spatiotemporal Sequences," *IEEE Trans. Knowledge and Data Eng.*, vol. 19, no. 4, pp. 453-467, Apr. 2007.
[2] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory Pattern Mining," *Proc. ACM SIGKDD*, pp. 330-339, Aug. 2007.
[3] G. Gidófalvi and T.B. Pedersen, "Mining Long, Sharable Patterns in Trajectories of Moving Objects," *GeoInformatica*, vol. 13, no. 1, pp. 27-55, 2009.

[4] J. Gudmundsson and M.J. Kreveld, "Computing Longest Duration Flocks in Trajectory Data," *Proc. 14th ACM Int'l Symp. Geographic Information Systems,* pp. 35-42, Nov. 2006.
[5] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory Clustering: A Partition-and-Group Framework," *Proc. ACM SIGMOD,* pp. 593-604, June 2007.
[6] J.-G. Lee, J. Han, and X. Li, "Trajectory Outlier Detection: A Partition-and-Detect Framework," *Proc. 24th Int'l Conf. Data Eng.,* pp. 140-149, Apr. 2008.
[7] J.-G. Lee, J. Han, X. Li, and H. Gonzalez, "*TraClass*: Trajectory Classification Using Hierarchical Region-Based and Trajectory-Based Clustering," *Proc. VLDB Endowment,* vol. 1, no. 1, pp. 1081-1094, 2008.
[8] X. Li, J. Han, S. Kim, and H. Gonzalez, "ROAM: Rule- and Motif-Based Anomaly Detection in Massive Moving Object Data Sets," *Proc. SIAM Int'l Conf. Data Mining,* Apr. 2007.
[9] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D.W. Cheung, "Mining, Indexing, and Querying Historical Spatiotemporal Data," *Proc. ACM SIGKDD,* pp. 236-245, Aug. 2004.
[10] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining Interesting Locations and Travel Sequences from GPS Trajectories," *Proc. 18th Int'l Conf. World Wide Web,* pp. 791-800, Apr. 2009.
[11] R. Fraile and S.J. Maybank, "Vehicle Trajectory Approximation and Classification," *Proc. Ninth British Machine Vision Conf.,* pp. 832-840, Sept. 1998.
[12] J. Krumm and E. Horvitz, "Predestination: Inferring Destinations from Partial Trajectories," *Proc. Eighth Int'l Conf. Ubiquitous Computing,* pp. 243-260, Sept. 2006.
[13] D.J. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H.A. Kautz, "Opportunity Knocks: A System to Provide Cognitive Assistance with Transportation Services," *Proc. Sixth Int'l Conf. Ubiquitous Computing,* pp. 433-450, Sept. 2004.
[14] H. Cheng, X. Yan, J. Han, and C.-W. Hsu, "Discriminative Frequent Pattern Analysis for Effective Classification," *Proc. 23rd Int'l Conf. Data Eng.,* pp. 716-725, Apr. 2007.
[15] H. Cheng, X. Yan, J. Han, and P.S. Yu, "Direct Discriminative Pattern Mining for Effective Classification," *Proc. 24th Int'l Conf. Data Eng.,* pp. 169-178, Apr. 2008.
[16] M. Deshpande, M. Kuramochi, N. Wale, and G. Karypis, "Frequent Substructure-Based Approaches for Classifying Chemical Compounds," *IEEE Trans. Knowledge and Data Eng.,* vol. 17, no. 8, pp. 1036-1050, Aug. 2005.
[17] C.S. Leslie, E. Eskin, and W.S. Noble, "The Spectrum Kernel: A String Kernel for SVM Protein Classification," *Proc. Seventh Pacific Symp. Biocomputing,* pp. 566-575, Jan. 2002.
[18] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text Classification Using String Kernels," *J. Machine Learning Research,* vol. 2, pp. 419-444, 2002.
[19] V.N. Vapnik, *Statistical Learning Theory.* John Wiley & Sons, 1998.
[20] G. Cong, K.-L. Tan, A.K.H. Tung, and X. Xu, "Mining Top-K Covering Rule Groups for Gene Expression Data," *Proc. ACM SIGMOD,* pp. 670-681, June 2005.
[21] W. Li, J. Han, and J. Pei, "CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules," *Proc. First IEEE Int'l Conf. Data Mining,* pp. 369-376, Nov. 2001.
[22] B. Liu, W. Hsu, and Y. Ma, "Integrating Classification and Association Rule Mining," *Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining,* pp. 80-86, Aug. 1998.
[23] X. Yin and J. Han, "CPAR: Classification Based on Predictive Association Rules," *Proc. Third SIAM Int'l Conf. Data Mining,* May 2003.
[24] R. Agrawal and R. Srikant, "Mining Sequential Patterns," *Proc. 11th Int'l Conf. Data Eng.,* pp. 3-14, Mar. 1995.
[25] X. Yan, J. Han, and R. Afshar, "CloSpan: Mining Closed Sequential Patterns in Large Databases," *Proc. Third SIAM Int'l Conf. Data Mining,* May 2003.
[26] H. Hu, D.L. Lee, and V.C.S. Lee, "Distance Indexing on Road Networks," *Proc. 32nd Int'l Conf. Very Large Data Bases,* pp. 894-905, Sept. 2006.
[27] R. Agrawal, T. Imielinski, and A.N. Swami, "Mining Association Rules between Sets of Items in Large Databases," *Proc. ACM SIGMOD,* pp. 207-216, May 1993.
[28] Y. Yang and J.O. Pedersen, "A Comparative Study on Feature Selection in Text Categorization," *Proc. 14th Int'l Conf. Machine Learning,* pp. 412-420, July 1997.
[29] Y.-W. Chen and C.-J. Lin, "Combining SVMs with Various Feature Selection Strategies," *Feature Extraction: Foundations and Applications,* I. Guyon, S. Gunn, M. Nikravesh, and L.A. Zadeh, eds., pp. 315-323, Springer, 2006.
[30] V. Sindhwani, P. Bhattacharya, and S. Rakshit, "Information Theoretic Feature Crediting in Multiclass Support Vector Machines," *Proc. First SIAM Int'l Conf. Data Mining,* Apr. 2001.
[31] J. Han and M. Kamber, *Data Mining: Concepts and Techniques,* second ed. Morgan Kaufmann, 2006.
[32] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.
[33] T. Brinkhoff, "A Framework for Generating Network-Based Moving Objects," *GeoInformatica,* vol. 6, no. 2, pp. 153-180, 2002.
[34] L.R. Rabiner and B.H. Juang, "An Introduction to Hidden Markov Models," *IEEE ASSP Magazine,* vol. 3, no. 1, pp. 4-16, Jan. 1986.
[35] P. Geurts, "Pattern Extraction for Time Series Classification," *Proc. Fifth European Conf. Principles of Data Mining and Knowledge Discovery,* pp. 115-127, Sept. 2001.
[36] E.J. Keogh and M.J. Pazzani, "An Enhanced Representation of Time Series Which Allows Fast and Accurate Classification, Clustering and Relevance Feedback," *Proc. Fourth Int'l Conf. Knowledge Discovery and Data Mining,* pp. 239-243, Aug. 1998.
[37] S. Gaffney and P. Smyth, "Trajectory Clustering with Mixtures of Regression Models," *Proc. Fifth ACM SIGKDD,* pp. 63-72, Aug. 1999.

Jae-Gil Lee received the PhD degree from Korea Advanced Institute of Science and Technology (KAIST) in 2005. He was with IBM Almaden Research Center, working on design and development of a data warehouse product. He is now an assistant professor in the Department of Knowledge Service Engineering at KAIST. He was a postdoctoral fellow at the Department of Computer Science, University of Illinois at Urbana-Champaign. His primary research interests include data mining, data warehousing, and database systems. His trajectory mining papers published at SIGMOD, VLDB, and ICDE are being actively cited in recent years. He received the Best Demonstration Award at ICDE '05 and served as the Publicity Chair of CIKM '09. He is a member of the IEEE.

Jiawei Han is a professor in the Department of Computer Science at the University of Illinois. He has been working on research into data mining, data warehousing, stream data mining, spatiotemporal and multimedia data mining, information network analysis, text and web mining, and software bug mining, with more than 400 conference and journal publications. He has chaired or served in more than 100 program committees of international conferences and workshops and also served or is serving on the editorial boards for *Data Mining and Knowledge Discovery*, the *IEEE Transactions on Knowledge and Data Engineering*, the *Journal of Computer Science and Technology*, and the *Journal of Intelligent Information Systems*. He is currently the founding editor-in-chief of the *ACM Transactions on Knowledge Discovery from Data* (TKDD). He has received IBM Faculty Awards, the Outstanding Contribution Award at the International Conference on Data Mining (2002), the ACM SIGKDD Innovation Award (2004), and the IEEE Computer Society Technical Achievement Award (2005). His book *Data Mining: Concepts and Techniques* (Morgan Kaufmann) has been used worldwide as a textbook. He is a fellow of the ACM and the IEEE.

Xiaolei Li received the BS, MS, and PhD degrees in computer science from the University of Illinois at Urbana-Champaign, in 2002, 2004, and 2008, respectively. He is currently working for Microsoft adCenter researching a variety of topics related to advertising and other online services.

Hong Cheng received the PhD degree from the University of Illinois at Urbana-Champaign in 2008. She is an assistant professor in the Department of Systems Engineering and Engineering Management at the Chinese University of Hong Kong. Her primary research interests include data mining, machine learning, and database systems. She has published more than 30 research papers in international conferences and journals, including SIGMOD, VLDB, SIGKDD, ICDE, ICDM, SDM, the *IEEE Transactions on Knowledge and Data Engineering*, the *ACM Transactions on Knowledge Discovery from Data*, and the *Data Mining and Knowledge Discovery*, and received research paper awards at ICDE '07, SIGKDD '06, and SIGKDD '05. She is a finalist for the 2009 SIGKDD Doctoral Dissertation Award.