PerNav: A Route Summarization Framework for Personalized Navigation

Yaguang Li[†], Han Su[†], Ugur Demiryurek[†], Bolong Zheng[‡], Kai Zeng[*], Cyrus Shahabi[†]

[†] University of Southern California, United States [*] Microsoft, United States [‡]University of Queensland, Australia
[†]{yaguang, hansu, demiryur, shahabi}@usc.edu [‡] b.zheng@uq.edu.au [*] kaizeng@microsoft.com

ABSTRACT

In this paper, we study a route summarization framework for Personalized Navigation dubbed *PerNav* - with which the goal is to generate more intuitive and customized turn-by-turn directions based on user generated content. The turn-by-turn directions provided in the existing navigation applications are exclusively derived from underlying road network topology information i.e., the connectivity of nodes to each other. Therefore, the turn-by-turn directions are simplified as metric translation of physical world (e.g. distance/time to turn) to spoken language. Such translation - that ignores human cognition about the geographic space - is often verbose and redundant for the drivers who have knowledge about the geographical areas. *PerNav* utilizes wealth of user generated historical trajectory data to extract namely "landmarks" (e.g., point of interests or intersections) and frequently visited routes between them from the road network. Then this extracted information is used to obtain cognitive turn-by-turn directions customized for each user.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial databases and GIS

1. INTRODUCTION

Navigation applications that find optimal routes and corresponding turn-by-turn directions in road networks are one of the fundamental and most used applications in wide variety of domains. While the problem of computing optimal path has been extensively studied and many efficient techniques have been developed over the past several decades, the turn-by-turn direction computation techniques have not changed. Meanwhile, with the ever-growing usage of navigation applications in mobile devices and car-navigation systems, plethora of user generated trajectory data became available. This data is particularly useful for producing more effective and cognitive turn-by-turn information.

SIGMOD'16, June 26-July 01, 2016, San Francisco, CA, USA
 ISBN 978-1-4503-3531-7/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2882903.2899384

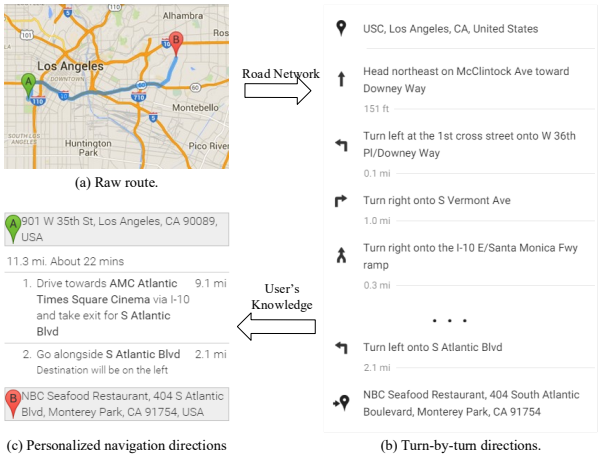

Figure 1: Personalized Navigation Directions.

Taken Figure 1 as an example, suppose we plan to go to a restaurant for dinner. Figure 1(a) shows a route provided by navigation systems. With road network information, turn-by-turn navigation directions can be generated (Figure 1(b)). These turn-by-turn directions are computed by taking into account inherent cost measure (e.g., distance or travel time) and angle (e.g., left or right) between the nodes of the underlying road network. While verbose, it is hard to be laconic without losing route information if we only have road network data. However, experienced urban commuters (i.e., majority of the drivers on roads) have good knowledge about the city and will probably be familiar with a certain part, sometimes most parts, of the route, e.g., routes from home to nearby highways. Suppose the navigation software knows that a user frequently visit a cinema which shares a similar route with the restaurant, it can replace a large part of navigation instructions with a single sentence, e.g., in Figure 1(c), drive towards the cinema via I-10 (the name of a highway). Then for the parts of the route that are unfamiliar to the user, navigation software may continue using detailed turn-by-turn directions. In this way, by leveraging the wealth of user's historical trajectory data, we may be able to generate navigation directions that are both concise and intuitive.

To achieve this goal, however, we have to further address challenges in multiple spheres. First, how to effectively extract landmarks and routes from trajectories which

usually come with noise and uncertainty? Second, how to utilize knowledge routes to generate personal navigation directions that are concise but convey enough information for user to interpret the route? - To address these challenges, we designed a prototype system called *PerNav* that produces turn-by-turn directions by leveraging the wealth of user's historical trajectory data. In particular, *PerNav* first extracts user's knowledge about the road network by leveraging trajectory calibration and trajectory clustering. Then for a given route, *PerNav* partitions it based on extracted knowledge using dynamic programming and branch-and-bound based approaches. The final step is to construct a personalized navigation directions based on the route partition. In summary, *PerNav* improves navigation applications in three ways. First, *PerNav* incorporates historical trajectory data to navigation applications to produce higher-quality personalized turn-by-turn directions for urban commuters. To our knowledge, this paper is the first to describe the use of trajectory data for generating personalized turn-by-turn directions. Second, *PerNav* offers network bandwidth reduction by significantly reducing the information transferred between the server and navigation clients. Third, *PerNav* allows user to customize the summary construction process with different detailed granularities.

The remainder of this paper is organized as follows. In Section 2, we formalize the problem and describe key techniques. In Section 3, we present our prototype system with real-world use cases. In Section 4, we review related work.

2. OVERVIEW

By referring to places and routes that are familiar to people, we can provide them an intuitive view about a new route. *PerNav* follows the same intuition to construct a higher level summary for a route generated by navigation software.

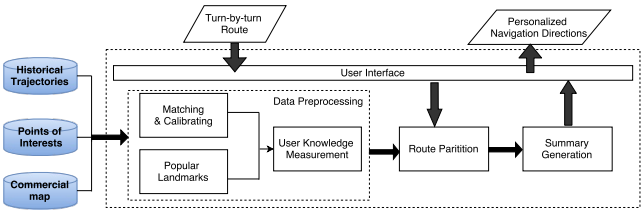

Figure 2: Framework overview of *PerNav*.

Figure 2 shows the overview of *PerNav*. The input of *PerNav* is a route and the output is a personalized navigation direction. This procedure mainly contains three steps, i.e., data preprocessing, route partition and summary generation. The first step for data preprocessing is offline while the last two steps are online. In the data preprocessing, *PerNav* will extract frequent visiting ⟨starting point, destination⟩ pairs and routes between them for every user, which are called the *knowledge* of users, from users' historical trajectories. Then given a route, *PerNav* partitions it based on extracted knowledge and the detail will be discussed in Section 2.3. The final step is to construct personalized navigation directions based on the route partition. In what follows, we will formalize the problem and describe the key technicals behind each step.

2.1 Preliminaries Concepts

We adopt the standard road network representation in which intersections are represented by nodes and roads are represented by links with attached features. Link is also known as road segment and we will use link and road segment interchangeably whenever the context is clear.

DEFINITION 1 (LANDMARK). *A landmark l is a geographical point in the space, which is stable and independent of user trajectories.*

A landmark can be either a Point of Interest (POI) or an intersection in the road network, and a route in the road network can be represented using landmarks.

DEFINITION 2 (ROUTE). *Formally, a route $\mathcal{R}$ in the road network is defined as a sequence of landmarks. $\mathcal{R} = [l_1, l_2, \cdots l_n]$. $\mathcal{R}(i)$ denotes the ith landmark, i.e., l_i. Every two adjacent landmarks are directly connected by a link in the road network. Thus $\mathcal{R}$ can also be considered as a sequence of links.*

In the road network, adjacent links usually share some features, e.g., street name, direction. Based on these features, we can divide the road network into disjointed routes, i.e., all the adjacent links with same features will be in the same route. As this type of route is independent of specific person, we call it *natural route* ($\mathcal{NR}$). For example, in Figure 1, S Atlantic Blvd is called a natural route. For a given route $\mathcal{R} = [l_1, l_2, \cdots l_n]$, we can use a sequence of natural routes $[\mathcal{NR}_1, \mathcal{NR}_2, \cdots, \mathcal{NR}_m]$ $(m \leq n)$, to represent it, which is known as the turn-by-turn manner, e.g., Figure 1(b), and this manner is widely used by existing navigation systems.

Moreover, there exists another type of route which is specific to individuals. In real life, urban commuters will get quite familiar with certain landmarks and routes between some of them, e.g., the route from home to work or the route from work to a shopping center. We call such a route *knowledge route* ($\mathcal{KR}$) for the urban commuter. In Figure 1(a), the route from home to the cinema is an example of knowledge route for the user.

Note that, a knowledge route usually contains multiple natural routes, i.e., $\mathcal{KR} = [\mathcal{NR}_i, \mathcal{NR}_{i+1}, \cdots, \mathcal{NR}_j]$. Thus, if this sequence of natural routes is a sub-sequence of $\mathcal{R}$, we can represent the route as:

$$\mathcal{R} = [\mathcal{NR}_1, \cdots, \mathcal{NR}_{i-1}, \mathcal{KR}, \mathcal{NR}_{j+1}, \cdots, \mathcal{NR}_k] \; k \leq m \leq n$$

which is called the personalized manner. Obviously, by using knowledge routes, the personalized manner becomes more concise, i.e., less number of the route-segments, and more familiar to the urban commuter than the turn-by-turn manner. In the following subsection, we will show how to extract and measure knowledge routes by analyzing user's historical trajectories.

2.2 User Knowledge Measurement

With historical trajectories, *PerNav* first employs trajectory calibration proposed in our previous research to map them into routes in the road network [8, 10]. Then stay points detection [15] is used to identify places that are familiar to user. These places are used to generate the description of knowledge routes. After that, *PerNav* groups trajectories into clusters [13]. In each cluster, the most representative route is extracted as a knowledge route connecting a source and a destination. Besides, for each knowledge route, we

calculate a familiarity score $f(\mathcal{R}) \in [0,1]$, which indicates the user's familiarity of this knowledge route. $f(\mathcal{R})$ is affected by the following aspects: 1) The frequency this route is traversed. 2) The significance of start and end landmarks of this route. The significance of landmarks are calculated by leveraging a HITS-like algorithm [15] which models the travellers as authorities, landmarks as hubs, and check-ins as hyperlinks. In *PerNav*, normalized weighted sum of these two factors is used as the familiarity score. Note that, for a natural route, its familiarity to a user is 0 by default.

Given the familiarity of a route $\mathcal{R}$, we can derive the familiarity of its segments, i.e., *route segment*.

Definition 3 (Route Segment). *A* route segment $\overline{RS} = \mathcal{R}(j,k) = [l_j, l_{j+1}, \cdots, l_k]$, *is defined as a sub-sequence of route* $\mathcal{R}$. *It can also be seen as a sequence of links connecting* l_j *and* l_k.

Based on the type of $\mathcal{R}$, a route segment can be either a *natural route segment* or a *knowledge route segment*. The familiarity of a route segment is calculated using Equation 1

$$f(\overline{RS}) = f(\mathcal{R}(\overline{RS})) \cdot g\Big(\frac{len(\overline{RS})}{len(\mathcal{R}(\overline{RS}))}\Big) \tag{1}$$

where $\mathcal{R}(\overline{RS})$ is the corresponding knowledge route of $\overline{RS}$, $\frac{len(\overline{RS})}{len(\mathcal{R}(\overline{RS}))}$ is the length ratio of $\overline{RS}$. $g(\cdot)$ is a monotonic convex function with $g(0) = 0$ and $g(1) = 1$. With the decrease of length ratio, the familiarity score decreases faster than a linear function which is more consistent with human being's cognition.

2.3 Knowledge-based Route Partition

Usually, if we want to describe a route, we will partition it into route segments and then give description for each of them.

Definition 4 (Route Partition). *For a route* $\mathcal{R}$, *a partition of it* $\mathbb{P}(\mathcal{R}) = \{\overline{RS}_1, \overline{RS}_2, \cdots, \overline{RS}_n\}$ *is such that:*

- $\bigcup_{i=1}^{n} \overline{RS}_i = \mathcal{R}$
- $\forall i, j\ \overline{RS}_i \cap \overline{RS}_j = \emptyset$

Each route segment $\overline{RS}_i$ *can be either a knowledge route segment or a natural route segment.*

For a route $\mathcal{R}$ in the road network, we can partition it in many ways resulting in possibly different navigation directions. Generally, we want the generated directions 1) be intuitive and easy for user to understand, i.e., utilizing knowledge routes with high familiarity in generated directions; and then 2) be concise, so it will be easy for human beings to read and share, i.e., minimize the number of route segments. With these objectives, we can evaluate the quality of a partition $\mathbb{P}$ by equation 2:

$$\mathcal{Q}_{\mathbb{P}(\mathcal{R})} = \sum_{\overline{RS}_i \in \mathbb{P}(\mathcal{R})} f(\overline{RS}_i) - \lambda |\mathbb{P}(\mathcal{R})| \tag{2}$$

$$\mathbb{P}_{opt}(\mathcal{R}) = \underset{\mathbb{P}(\mathcal{R})}{\operatorname{argmax}}\, \mathcal{Q}_{\mathbb{P}(\mathcal{R})} \tag{3}$$

where $f(\cdot)$ is the function to calculate the familiarity score in Equation 1, $|\mathbb{P}(\mathcal{R})|$ is the number of route segments in this partition, $\lambda \geq 0$ is used to constrain the number of segments in the generated route partition. We want to find the route partition $\mathbb{P}(\mathcal{R})_{opt}$ that maximize $\mathcal{Q}_{\mathbb{P}(\mathcal{R})}$.

The naïve approach is to enumerate all the possible combination of route segments and choose the one with maximum score. However, the time complexity is exponential in terms of the number of links in the route. In the following paragraphs, we will show that dynamic programming can be used to find the optimal route partition efficiently.

Definition 5 (Conditional Route Partition). *Given a route* $\mathcal{R} = [l_1, l_2, \cdots, l_n]$, *its conditional route partition* $\mathbb{P}(\mathcal{R}(1,i)|\mathcal{R}'(\cdot,\cdot))$ *is defined as route partitions with the last route segment, i.e.,* $\overline{RS}$, *coming from route* $\mathcal{R}'$.

Specially, $\mathbb{P}\big(\mathcal{R}(1,i)|\mathcal{R}'(j,k)\big)$ represents the conditional route partition with the last route segment equals to $\mathcal{R}'(j,k)$.

Lemma 1. *The optimal partition of a route is the best conditional optimal partition among all candidate routes for the last link.*

$$\mathbb{P}_{opt}(\mathcal{R}(1,i)) = \underset{\mathbb{P}}{\operatorname{argmax}}\, \mathcal{Q}\left(\mathbb{P}_{opt}(\mathcal{R}(1,i)|\mathcal{R}'(\cdot,\cdot)\right) \tag{4}$$
$$\mathcal{R}' \in \mathbb{R}(\mathcal{R}(i-1,i))$$

where $\mathbb{R}(\mathcal{R}(i-1,i))$ is the set of all the routes that contains link $\mathcal{R}(i-1,i)$. The proof is straightforward, as we enumerate all possible conditions and choose the best one.

Lemma 2. *The conditional optimal route partition of a sub-route* $\mathcal{R}(1,i+1)$, *i.e.,* $\mathbb{P}_{opt}(\mathcal{R}(1,i+1)|\mathcal{R}'(\cdot,\cdot))$, *can be derived from conditional optimal route partitions of* $\mathcal{R}(1,i)$, *i.e.,* $\mathbb{P}_{opt}\big(\mathcal{R}(1,i)|\mathcal{R}''(\cdot,\cdot)\big)$.

$$\mathbb{P}_{opt}(\mathcal{R}(1,i+1)|\mathcal{R}') = \underset{\mathcal{R}'' \in \mathbb{R}(\mathcal{R}(i-1,i))}{\operatorname{argmax}} \tag{5}$$
$$\mathcal{Q}\Big(\mathbb{P}_{opt}\big(\mathcal{R}(1,i)|\mathcal{R}''(\cdot,\cdot)\big) \cup \big(\mathcal{R}(i,i+1) \mapsto \mathcal{R}'\big)\Big)$$

where $\mathbb{R}\big(\mathcal{R}(i-1,i)\big)$ is the set of all the routes that contain link $\mathcal{R}(i-1,i)$, and $\mathbb{P}_{opt}\big(\mathcal{R}(1,i)|\mathcal{R}''\big) \cup \big(\mathcal{R}(i,i+1) \mapsto \mathcal{R}'\big)$ means mapping the link $\mathcal{R}(i,i+1)$ to $\mathcal{R}'$ and merge it to the conditional optimal partition $\mathbb{P}_{opt}(\mathcal{R}(1,i)|\mathcal{R}'')$. Let $\Delta\mathcal{Q}\big(\mathcal{R}(i,i+1) \mapsto \mathcal{R}'\big)$ denotes the score change after mapping the link $\mathcal{R}(i,i+1)$ to $\mathcal{R}'$. According to Equation 2, $\Delta\mathcal{Q}\big(\mathcal{R}(i,i+1) \mapsto \mathcal{R}'\big)$ will be the same as long as the last segment of $\mathbb{P}$ is mapped to a route other than $\mathcal{R}'$. Thus, we can iteratively find the optimal partition using dynamic programming.

2.4 Summary Construction

After the partitioning step, a route $\mathcal{R}$ will become a sequence of route segments. Then *PerNav* will generate navigation directions by describing knowledge route segments and natural route segments respectively. As knowledge route $\mathcal{R}$ is familiar to user, we can describe it using some key features, e.g., the start/end landmark, the main highway, rather than describing all the natural routes in it. In order to construct more fluent summary sentences, we also define sentence templates, e.g., drive towards *POI name* via *most representative feature of the knowledge route*. The most significant POI within a certain distance is selected to generate the summary.

For natural routes, traditional turn-by-turn directions generations techniques can be used. Moreover, in *PerNav* user can specify the description granularity of directions. Under the configuration of the most detailed granularity, *PerNav* will generate descriptions for every single route segments.

The result will be similar to turn-by-turn directions provided by current navigation systems. With the decrease of detail level, less important route segments, i.e., low-level roads and short roads, will be omitted.

3. DEMONSTRATION

In this section, we will present a detailed demonstration plan and how user can interact with the system. *PerNav* has both web application and mobile application. Figure 3 shows GUI of the web application.

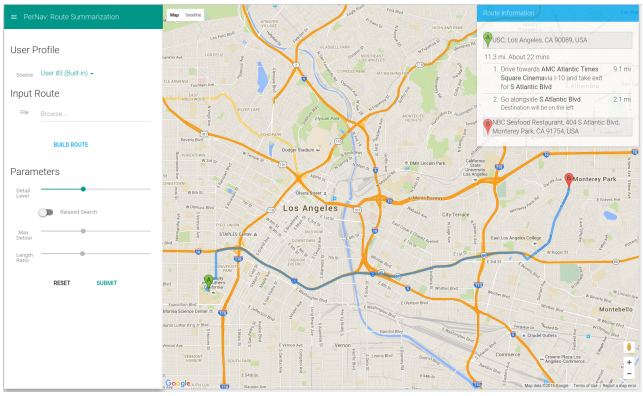

Figure 3: Main GUI

The left panel is for settings and query setup while the right panel is used for result visualization. Since *PerNav* requires historical trajectories information to work properly, we create several built-in profiles based on the following trajectory datasets: Geolife [14, 15], and Planet.gpx [1]. Geolife dataset consists of trajectories of 178 users over a period of four years while Planet.gpx contains the GPS traces uploaded by OpenStreetMap users within 7.5 years. Users can click to select which profile they prefer. Besides, users are allowed to create their own profiles. This can be done by specifying a set of ⟨source, destination⟩ pairs, a route connecting each pair and familiarity score of each route.

After choosing the profile, user can construct the input in following ways: 1) *PerNav* provides several built-in query routes for demonstration purpose; 2) user can interactively create a route by pinning and dragging. Then *PerNav* will generate a personalized route description with corresponding details visualized on the map. In addition, users are allowed to play with a set of parameters to customize the route summarization procedure.

4. RELATED WORK

The problem proposed in this work is relevant to trajectory processing and trajectory querying issues, including trajectory data mining [13, 5, 6], popular route discovery [7, 3], personalized route recommendation [2, 4, 9]. The method to discover personalized routes from trajectories is proposed in [2], while the approach to find popular routes is investigated in [3]. In [12, 4], algorithms to recommend routes based on historical trajectories are described. [11] proposes a system to construct summary for raw trajectory. However, none of these problems are same with ours. Our work is mainly regarding reinterpreting an existing route in a way that is intuitive to user and easy to follow, while the vast majority of existing works are dealing with general mining/prediction/recommendation problems.

Acknowledgment

This research has been funded by the USC Integrated Media Systems Center (IMSC) and unrestricted cash gifts from Oracle and Northrop Grumman Corporation.

5. REFERENCES

[1] OpenStreetMap. http://wiki.openstreetmap.org/wiki/Planet.gpx.

[2] K.-P. Chang, L.-Y. Wei, M.-Y. Yeh, and W.-C. Peng. Discovering personalized routes from trajectories. In *ACM SIGSPATIAL Workshop on Location-Based Social Networks*, pages 33–40. ACM, 2011.

[3] Z. Chen, H. T. Shen, and X. Zhou. Discovering popular routes from trajectories. In *ICDE*, pages 900–911. IEEE, 2011.

[4] D. Delling, A. V. Goldberg, M. Goldszmidt, J. Krumm, K. Talwar, and R. F. Werneck. Navigation made personal: Inferring driving preferences from gps traces. In *ACM GIS*. ACM, 2015.

[5] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *SIGKDD*, pages 330–339. ACM, 2007.

[6] J.-G. Lee, J. Han, X. Li, and H. Gonzalez. Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. *Proceedings of the VLDB Endowment*, 1(1):1081–1094, 2008.

[7] X. Li, J. Han, J.-G. Lee, and H. Gonzalez. Traffic density-based discovery of hot routes in road networks. In *SSTD*, pages 441–459. Springer, 2007.

[8] Y. Li, C. Liu, K. Liu, J. Xu, F. He, and Z. Ding. On efficient map-matching according to intersections you pass by. In *DEXA*, pages 42–56. Springer, 2013.

[9] S. Shang, R. Ding, B. Yuan, K. Xie, K. Zheng, and P. Kalnis. User oriented trajectory search for trip recommendation. In *EDBT*, pages 156–167. ACM, 2012.

[10] H. Su, K. Zheng, H. Wang, J. Huang, and X. Zhou. Calibrating trajectory data for similarity-based analysis. In *SIGMOD*, pages 833–844. ACM, 2013.

[11] H. Su, K. Zheng, K. Zeng, J. Huang, and X. Zhou. STMaker: a system to make sense of trajectory data. *VLDB*, 7(13):1701–1704, 2014.

[12] J. Yuan, Y. Zheng, X. Xie, and G. Sun. Driving with knowledge from the physical world. In *SIGKDD*, pages 316–324. ACM, 2011.

[13] Y. Zheng. Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):29, 2015.

[14] Y. Zheng, X. Xie, and W.-Y. Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[15] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from gps trajectories. In *World Wide Web*, pages 791–800. ACM, 2009.