Advance Access publication on 23 April 2015
doi:10.1093/comjnl/bxv030

Mining Large-Scale GPS Streams for Connectivity Refinement of Road Maps

YANMIN ZHU[1,2*], YIN WANG[3], GEORGE FORMAN[4] AND HONG WEI[1]

[1]*Shanghai Key Lab of Scalable Computing and Systems, Shanghai, China*
[2]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China*
[3]*Facebook, Menlo Park, USA*
[4]*HP Labs, Palo Alto, USA*
[*]*Corresponding author: yzhu@cs.sjtu.edu.cn*

As people increasingly rely on road maps in the digital age, manually maintained maps cannot keep up with the demand for accuracy and freshness, evidenced by the recent iOS map incident and the bidding war for Waze. There are many research works on automatic map inference using GPS data, and some have suggested that Google and Waze automate their map update processes to some degree with user data. However, existing *published* work focuses on refining road geometry. In reality, connectivity issues at intersections, including missing connections and unmarked turn restrictions, are much more prevalent and also more difficult to infer. In this paper, we report our study on the connectivity issues in the OSM Shanghai map using 21 months of GPS data from over 10 000 taxis. We first adapt a robust *map matching* algorithm. Then, we propose simple algorithms for detecting missing intersections, turn restrictions and road closures. Empirical results show that our algorithms of connectivity refinement for road maps are effective.

Keywords: spatial data mining; GPS; map inference; road maps

Received 26 June 2014; revised 21 March 2015
Handling editor: Fionn Murtagh

1. INTRODUCTION

As people increasingly rely on navigation using smartphones for everyday activities, inaccurate maps have substantial economic consequences and even threaten safety, evidenced by the recent quality problem of iOS 6 maps, and the subsequent bidding war for Waze. Car accidents caused by or related to digital maps are frequently reported in the news [1–6]. A British insurance survey found 26% of drivers had been directed by their GPS to go through no-entry signs or prohibited areas [7]. Washington State alone reported 623 collisions in a 4-year period caused partly by GPS issues, two of which were fatal [8]. The root cause of the map quality problem is the manual map creation and update process, which is costly, error prone and cannot keep up with the aggregated rate of changes to the entire road network.

In response to the challenge, there is a significant amount of work on automated map inference and update, typically using either aerial imagery [9, 10] or GPS data [11–14]. The use of aerial imagery for road recognition is more effective in identifying highways; recognizing narrow local roads is very challenging, especially those covered by tree canopy. And it is ineffectual for detecting most turn restrictions. Further, its freshness depends on the image update cycle, which tends to be infrequent. It has been reported that Google maps is working to use image recognition techniques to extract road metadata information from street views [15].

Map inference from GPS traces is gaining attention because of the increasingly abundant and fresh data from smartphones. Most of the existing works focus on algorithms that build new maps from GPS data; see [12, 13] and references therein. Our recent work proposes a map update system called CrowdAtlas, which corrects and updates existing maps using live GPS streams. CrowdAtlas first employs *map matching* [16] algorithms to break GPS traces into matched segments that are well aligned with the map, and unmatched segments that are discrepant with the existing map. Matched segments are then used to *refine* the map, and unmatched segments are used to *augment* the map.

CrowdAtlas focuses on updating roads, including geometry refinement and new road detection. In practice, we find that

intersection errors are more prevalent and are very difficult to detect, including erroneous or wholly missing intersections, as well as incorrect turn restrictions. For example, in the map matching competition of SIGSPATIAL Cup 12 [17], many teams discovered a discrepancy between a training trace and the map, shown in Fig. 1. Here the sequence of black dots represent the input GPS trace, for which the ground-truth drive path is the yellow-highlighted road followed by the horizontal blue-highlighted road. However, the provided map, which originates from the TIGER product of U.S. census bureau, erroneously does not connect the two roads. We will show that missing intersections is a common human error in OpenStreetMap (OSM) [18]. This type of error is almost impossible to detect without trace data, because the roads often touch each other or are extremely close, rendering visual review ineffective.

Turn restriction errors are more difficult to find than missing intersections, and they plague even commercial maps. For example, both Google and Bing maps have serious turn restriction errors near the intersection of Market St. and Freeway 101 in San Francisco, one of the busiest intersections of the city, which has caused fatal accidents in the past due to illegal turns [19]. There is almost no research on map connectivity inference from GPS trace data. A rare exception is [20], which detects the locations of road intersections, absent a base map. The problems of detecting turn restrictions and updating existing maps were not considered.

In this paper, we extend our map update framework [14] to address connectivity issues in existing road maps. First, we use unmatched traces to detect missing intersections and automatically add them to the map where sufficient GPS evidence has been collected. Then, we propose algorithms to use matched traces to detect turn restrictions and road closures. We evaluate our approach on 21 months of GPS data collected from thousands of taxis in Shanghai, using OSM as our base map. One month of the data are available online [21]. We verify long-term road closures using historical aerial imagery from Google Earth, as well as news sources. Of the 88 total closures our system detected that lasted more than 2 weeks, 80 of them were verified as correct; the rest are unverifiable, not false positives.

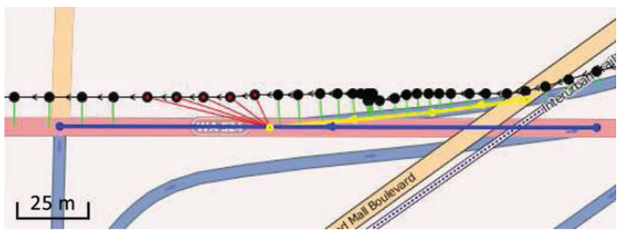

FIGURE 1. An example of map connectivity error in SIGSPATIAL Cup 12.

Contributions: This work is the first to use data mining of large-scale GPS traces in order to automatically correct and refine road intersections. We elucidate the value, requirements and practical challenges for this novel application. Finally, through extensive search and verification, we validated some of the missing intersections and closures that were identified.

Section 3.1 provides background and Section 3 describes our proposed methods, which are validated by the empirical results in Section 4. Section 5 discuss related work, and finally Section 6 concludes, with perspectives on future work.

2. BACKGROUND ON MAP UPDATE

Figure 2 is the simplified map update framework proposed in [14]. The incoming GPS streams first go through a *map matching* process, which intends to find the most likely routes on the existing map for each vehicle. This takes into account the geometry of the road network between consecutive GPS points, which may arrive many seconds apart to reduce data volume. (Fleet management applications do not need position fixes every second, unlike traditional GPS mapping applications.)

The customized map matching algorithm, detailed in the next subsection, is aware of map inaccuracy, and partitions the input GPS trace into matched and unmatched segments. Matched trace segments are those that align well with the map, within ∼50 m to the matched route. We derived the parameter of 50 m based on our empirical study [22]. We use these segments to refine the given map, e.g. adjust road geometry or detect erroneous turn restrictions. Unmatched trace segments are used to augment the map, e.g. adding new roads or missing intersections.

Map matching is crucial to the overall map update process, affecting both precision and recall of the update result. Next we discuss the background on map matching and our customized map-inaccuracy-aware matching algorithm.

2.1. Map matching

The input of a map matching algorithm is a sequence of GPS coordinates with the corresponding timestamps of a given vehicle, and the output is a sequence of GPS coordinates each

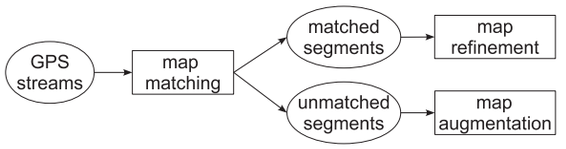

FIGURE 2. Map update using GPS streams.

of which either is on a road in the map or labeled as *not matched*.

Map matching algorithms can be largely divided into two categories, i.e. *incremental* and *global*. Incremental algorithms match one sample at a time, using only past observations, which is useful for navigation. Global algorithms match the entire trace at once, which in general achieves better accuracy by sacrificing freshness of the result. Our prior work contains references to many representative map matching algorithms in both categories and a performance comparison study [22]. In the map matching competition of SIGSPATIAL Cup 12 [17], the top two accurate programs both use global algorithms [22, 23] that are variants of the Hidden Markov Model (HMM)-based map matching [16].

We customized our prior HMM map matching algorithm [22] for the purpose of splitting matched and unmatched trace segments. Our experiments show that it achieves more than 95% accuracy for sampling intervals as sparse as 1 min [22], which is more than sufficient for map update using our taxi dataset. In the dataset, the GPS trace of a taxi consists of such data points as

$$\langle taxi\_ID, long., lati., heading, timestamp \rangle.$$

Because HMM-based map matching is a global algorithm, we need to break the incoming GPS stream of a vehicle into sections, e.g. by 5 min intervals. In practice, there are sufficient natural 'breaks' in the stream that can be used to our advantage, e.g. parking, waiting for a customer or a traffic light, and long intervals of silence due to lost satellite signal or cellular network connection. We use these natural breaks to split the GPS streams.

Our algorithm can detect two types of mismatches between GPS traces and road maps. Type I is when there is no road within the error radius (50 m) of a sample. Type II is when there are roads within the error radii of two consecutive samples, but the shortest path that connect any pair of them is too long, e.g. exceeding the maximum travel speed (180 km/h) multiplied by the time interval between the two samples. Type I mismatch typically corresponds to a new road, which we studied in [14]. We focus on the Type II mismatch in this paper, which typically corresponds to a missing intersection. We provide a definition of Type II mismatch as follows.

DEFINITION 2.1 (Type II mismatch). *Consider two consecutive samples s_{t_1} and s_{t_2} and they are mapped to roads. Let $s^*_{t_1}$ and $s^*_{t_2}$ denote the mapped points for s_{t_1} and s_{t_2}, respectively. We use p to denote the shortest path on the road map connecting $s^*_{t_1}$ and $s^*_{t_2}$. And let $l(p)$ denote the length of the path. A Type II mismatch exists if*

$$l(p) > \upsilon_0 \times (t_2 - t_1) \quad (1)$$

where υ_0 is the maximum travel speed of a vehicle.

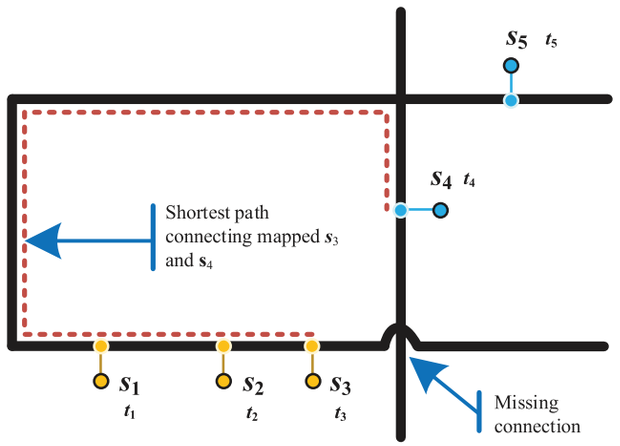

FIGURE 3. An example showing the mismatch of type II, in which there is a sequence of samples (s_1, s_2, s_3, s_4 and s_5). The samples are mapped to roads. Consider the two mapped samples, s_3 and s_4. Since originally the connection is missing, the shortest path connecting mapped s_3 and s_4 is shown, which is too large for a regular vehicle to travel within a short time.

Figure 3 illustrates the definition given above. Figure 1 shows a Type II mismatch in a real setting. Our customized map matching algorithm detects this type of mismatch because there is no route that can connect the yellow road to the blue road within the maximum speed the vehicle can travel.

3. SOLUTION

This section presents the design of our algorithms proposed for connectivity refinement of road maps.

3.1. overview

The core ideas of connectivity refinement are straightforward. Specifically, connectivity refinement consists of (1) *detecting missing intersections*, (2) *detecting road restrictions* and (3) *detecting road closures*. Intuitively, an intersection is missing between roads A and B if there is a trace that first matches to A and then to B. For every turn possibility of every intersection, if there is never any traffic, the turn is prohibited. If the traffic changes from none to a positive value, or vice versa, the turn becomes open, or closed, respectively.

It is, however, a non-trivial matter to reduce these simple precepts to effective practice, especially because this map update application, by its very nature, requires large-scale data input, which always exhibits a wide variety of real-world issues. First, the data source may have occasional days with missing data, or days having only a fraction of the usual volume (see Fig. 4). We would not want the software to declare all turns restricted because of a data outage or a major holiday.

Secondly, there is always noise in the data, including inherent GPS noise and drivers making illegal turns. This is

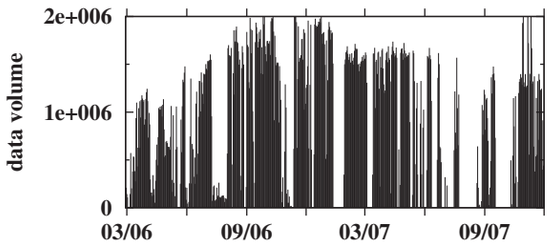

FIGURE 4. Highly variable data volume per day.

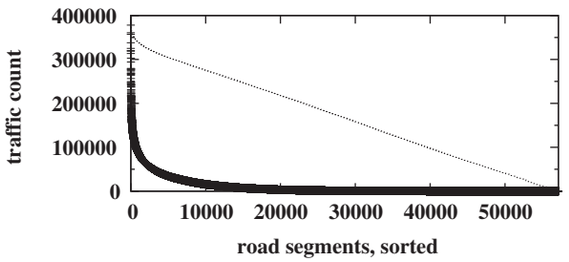

FIGURE 5. Highly skewed traffic per road.

especially true for intersections beside highways that carry a great volume of traffic nearby, especially if the highways are elevated or there are tall buildings present, whose satellite shadows and reflections increase the GPS location error. Based on this observation, it is natural to enforce a traffic threshold for intersection detection. This would not work well because of the next difficulty.

Thirdly, there is a great deal of disparity in data coverage of different types of roads and different areas of the map. The majority of roads in a city map receive small traffic rates. As with many natural phenomena, the distribution of traffic across roads exhibits high skew, as shown by the dark curve in Fig. 5, where each point represents the traffic volume on an individual road, sorted by frequency. (The thin line shows the log(count) scaled to fill the graph; its good linear fit reveals an exponential distribution). Because of this highly skewed distribution, techniques that can extend the effectiveness to slightly lower traffic rates can yield a super-linear increase in effective road coverage.

The main steps of our solution are as follows. First of all, we apply the map matching algorithm as decried in Section 2.1 on the GPS dataset. Our solution uses a dataset of GPS traces of vehicles (taxis in our empirical study). For each vehicle, the trace of a taxi is a sequence of GPS coordinates with the corresponding timestamps. As a result of this step, we have the dataset of map-matched GPS traces. Then, we select candidate intersections based on the digital map and then check if each of the candidate misses an intersection. Third, for each of all the turn possibilities in the road map, we check if it is prohibited.

FIGURE 6. Y-splits and Y-merges punctuate *closure segments*.

Finally, for each of the road sections in the road map, we check if it is closed for a given long time window.

3.2. Calculating turn ratios and aggregating road segments

To overcome the above-mentioned difficulties, we propose a metric called *turnRatio* and aggregate short road segments to longer closure roads.

One important building block for connectivity refinement is to look at the traffic on traversing road segments and from a road segment to a neighboring road segment. Specifically, we refer the traffic on a road segment to the number of traces that traversing through the road segment. One issue for looking merely at the traffic is that there is a significant difference of the traffic volumes of different road segments. To offset such difference, instead of considering the absolute traffic turning from road A to B, we calculate the ratio between the turning traffic and the minimum traffic passing on A and B. It is more formally defined as

$$turnRatio(A, B) = \frac{\text{traces from } A \text{ to } B}{\min(\text{traces passing } A, \text{traces passing } B)} \quad (2)$$

Because of data disparity, we consider the minimum traffic on A and B instead of average or other such measurements. It is quite common that a residential road is connected to a major road where the latter has at least an order of magnitude more traffic than the former.

By using the *turnRatio* metric, we can therefore distinguish noisy illegal turns at a busy intersection from normal turns at an infrequently traveled intersection, even though the absolute traffic volumes for both kinds of turns can be similar.

One problem with the turn ratio calculation method is of road length. If road A or B is too short and receives very little matched samples, the ratio calculation can be misleading. Next we propose a technique of road segment aggregation to address this challenge.

We aggregate road segments into coarser-grained *closure segments*, each of which is monitored as a unit for traffic counting. Figure 6 illustrates closure segments by color, coalescing many individual pieces of road segments in the digital

FIGURE 7. Aggregating roads into *closure segments*. Road segments *A*, *B* and *C* are aggregated into one closure segment, and road segments *D* and *E* are aggregated to a separate closure segment.

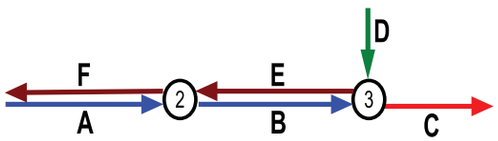

FIGURE 8. Example fragment of a digital map.

map, to which GPS samples are attached by map matching. Each closure segment connects two adjacent intersections. For example in the figure, Y-intersections punctuate separate closure segments. All GPS data points matched to a closure segment are included for the turn ratio calculation, which significantly increases the coverage for sparse data—the common case for fleet management systems—where each vehicle's location is not obtained every second and short road segments are therefore skipped in many traces.

Note that closure segments can sometimes aggregate portions of multiple, named roads. For example, in Fig. 7 the segments of roads *A*, *B* and *C* are combined into a common closure segment, and the road segments *D* and *E* are combined into a separate closure segment. Although *C* and *D* have the same road name and we normally think of them together as one 'road', with regard to intersections they need to be considered separately.

The algorithm for aggregating short road segments to longer closure segments is described as follows. To partition the road network into closure segments, it performs a breadth-first traversal of the graph of *directed* road segments. Two-way roads contribute segments in each direction, which improves the precision of map-matching [22] and allows us to consider every possible turn separately. During the traversal we assign consecutive road segments to the same closure segment until an intersection node of degree $\geq$3, where the degree is determined by the number of *undirected* roads attached to the intersection. That is, two-way roads only count once toward the undirected degree. In the example in Fig. 8, a traversal of segment *A* can lead only to segment *B*, thus we continue the same closure segment (blue) through the node of degree 2. But when arriving at the node with degree 3, there are multiple input paths (*B* and *D*); this three-road intersection ends the blue closure segment, and the traversal of *C* and *E* each begin separate closure segments. Traversal is initiated at nodes with degree $\geq$3.

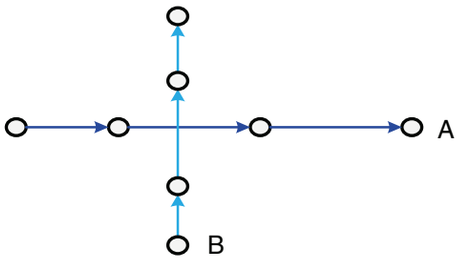

FIGURE 9. A missing intersection of type 1 between roads *A* and *B*.

3.3. Detecting missing intersections

The idea of our missing intersection detection algorithm is conceptually simple. There is a missing intersection between two disconnected roads if a trace matches to them in a sequence. The method proceeds in two steps.

In the first step, we derive the candidate pairs of disconnected roads based on the base digital map. In practice, with real-world large GPS datasets, traces connect numerous pairs of roads due to noise and sparse sampling [24]. Fortunately, we have a base digital map. In the digital map, roads are directed. Each road segment is represented by a sequence of vertices and directed edges connecting two neighboring vertices, as illustrated by Fig. 8. We only need to focus on two types of potential missing intersections. As illustrated by Fig. 9, a missing intersection of type 1 occurs between two unconnected roads that touch each other (i.e. cross each other) but do not share a vertex, and a missing intersection of type 2 occurs between two roads whose endpoints are extremely close to another road (e.g. within 5 m).[1] Although we consider only these two types of missing intersections, there are a significant number of them in a metropolis. For example in Beijing, there are 15 896 pairs of roads that touch each other, and 263 pairs of roads that are extremely close to other roads.

Figure 10 shows a complex highway interchange that includes at least eight potentially missing intersections (Fig. 11).

In the second step, for each candidate pair of roads, we check if it misses an intersection. The basic idea is to check if there are map-matched GPS traces that transverse from a road to the other disconnected road. To remove the influence of noisy traces, however, we use the metric of *turnRatio(A, B)* for a candidate pair of roads *A* and *B*. If the ratio exceeds a threshold and the passing traffic on *A* and *B* are sufficient, we determine that there is a missing intersection between *A*

[1] If GPS traces connect two *distant* roads, there is typically a missing road, which can be effectively detected using road inference algorithms [14].

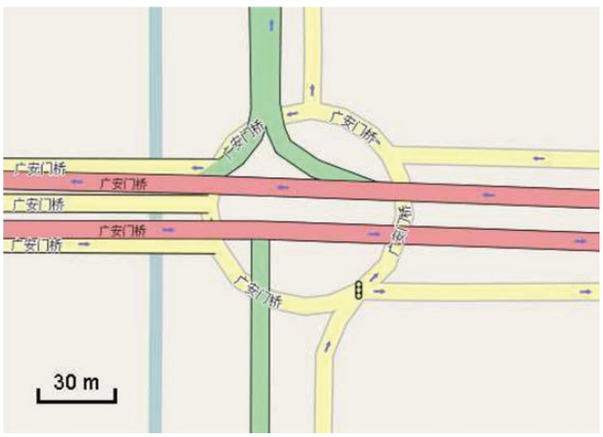

FIGURE 10. Need to consider every pair of disconnected roads that touch each other for missing intersection detection, at least eight pairs in this case.

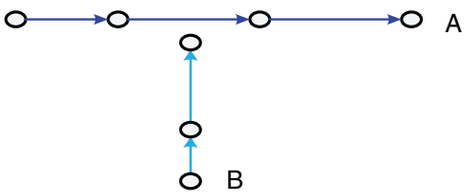

FIGURE 11. A missing intersection of type 2 between roads A and B.

and B. The latter requirement is to avoid infrequently traveled roads with only noise data. For example, some taxi drivers may take illegal shortcuts at construction sites at night. Note that the threshold on the turn ratio can be selected based on empirical study. It controls the tradeoff between false positives and false negatives. In our experiments, we set the threshold to 0.1.

Once a new intersection is detected between roads A and B, the problem remains to find the location of the intersection. For the first type of missing intersection, it is naturally located at the place where the two roads touch each other. For the second type, we extend the road whose endpoint is close to the other until it intersects.

3.4. Detecting turn restrictions

Detecting turn restrictions also proceeds in two steps. First, we obtain all possible turns by enumerating all road intersections. This is simple with the use of the digital map. Secondly, for every possible turn of every intersection, we use the matched GPS traces to detect turn restrictions. More specifically, we determine that there is a turn restriction from A to B if *turnRatio(A,B)* is less than a threshold, and there is sufficient traffic on both A and B. Here the latter requirement is crucial because the data coverage of different roads can be highly skewed (recall Fig. 5). We must not declare a turn restriction from A to B if there is simply very little traffic on either

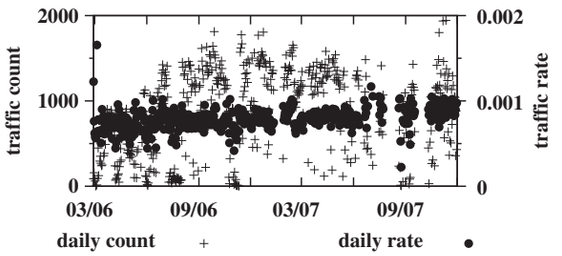

FIGURE 12. The count of traffic GPS traces traversing a closure segment in a day shows larger variation than the rate of the count of the segment to the total sum of the counts of all closure segments in the map.

road. Note again the threshold is set based on empirical study. In general, this threshold also strikes the tradeoff between the false positive rate and the false negative rate. In our experiments, we set the threshold to 0.005.

With continuous GPS streams, we must be cautious in declaring a permanent turn restriction, because there can be temporary closures due to accidents, road maintenance or sporting events. This is in contrast to missing intersection detection, where a new intersection is inserted right away with sufficient evidence (and its turn restrictions can be refined later). In general, we use GPS traces within a relatively long time window (e.g. 3 months) and apply the turn restriction detecting algorithm to detect if a turn is prohibited within this time window.

3.5. Detecting road closures

Road closures are relatively more dynamic over time. The basic idea of detecting the closure of a road is to check if the traffic traversing through the road dramatically drops from a significant volume to zero. Conversely, the reopening of a road is detected when the traffic traversing through the road raises from zero to a significant volume. To check the traffic change over time, we divide the time into slots of a fixed length, e.g. 1 day.

For each closure segment, we focus on the traces traversing the segment in each time slot. Similar to turn restriction detection, we consider the rate between the number of the traces through a closure segment and the total number of traffic through all closure segments to eliminate the slot-level variance. The effect of reducing the variance by using the rate is illustrated by Fig. 12. To detect the closure of a closure segment, we compute the moving average of the recent slots. A closure is detected if the moving average is close to zero. Note that the window size is an important parameter that needs to be tuned. The windows size controls the tradeoff between the ability to fight short traffic variance and the closure detection latency. Rather than triggering on any slot with a zero ratio, we identify closure segments having periods of $\geq$2 weeks that closed and remained so through the end of the dataset.

4. EMPIRICAL RESULTS

We use two taxi datasets for evaluation. For missing intersection detection, we use a relatively small dataset of 70 taxis from Beijing for 1 week in 2008. These taxis sampled at 10 s intervals, relatively short compared with other taxi datasets [24, 25], and convenient for verification. Turn restriction detection and road closure detection need datasets of longer durations. Therefore, we use 21 months of taxi traces from Shanghai for evaluation, 27 February 2006–30 November 2007. Both the Beijing dataset and 1 month of the Shanghai dataset are available online [19]. Altogether the Shanghai dataset has over 2 billion samples, 120 GB uncompressed. The amount of data varies each day, as shown in Fig. 4. Of the 642 days, 139 of them have no data at all. For the rest of the days, we typically get at least 2000 taxis reporting non-stationary traces. Different taxis have various sampling configurations. Most of them use a 1 min interval or longer, but there is also a large group of taxis that sample at 61 s when occupied and 16 s when vacant. Since our map matching algorithm maintains high accuracy even at long sampling intervals [22], we break input GPS streams only when the interval is beyond 3 min. After filtering and map matching, 827 983 408 out of the 2 billion samples are matched (40%).

We use OSM [18] as our base map. In China, freeways and truck roads come from a commercial map donation, while other roads are drawn by volunteers based on aerial imagery. We find that even the latest OSM maps of Beijing and Shanghai are far from complete, and earlier version proved lacking the required detail. Therefore, we use the map of 28 November 2012 as our base map, which may contain roads that did not actually exist when the data were collected. All roads of Beijing on this map are included in intersection detection, while *new* intersections are those not on this map but being used by the end of 2008, i.e. for which we have GPS data but not map data. For turn restrictions and road closures, all intersections of Shanghai on this map are included in the evaluation. More detailed analysis of the dataset and the map is available in [24].

4.1. Intersection detection

In Beijing, there are 15 896 pairs of roads that cross each other but are not connected, and 263 roads whose endpoints are within 5 m to other roads. Our Beijing dataset has traces that match to 57 pairs of type 1 and 5 pairs of type 2 of potential missing intersections. Among them, 43 of type 1 and 4 of type 2, respectively, are manually confirmed as real missing intersections using the Baidu map. Figure 13 shows all $57 + 5 = 62$ intersections and the counts of their support traces. A support trace is such a trace based on which the pair of the disconnected segments can be connected. If a single support trace is considered sufficient evidence, then there is a significant percentage of false positives. But by requiring at least three support traces, all false positives were eliminated.

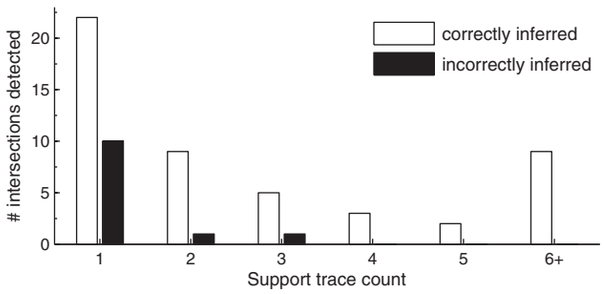

FIGURE 13. Missing intersections detected in Beijing.

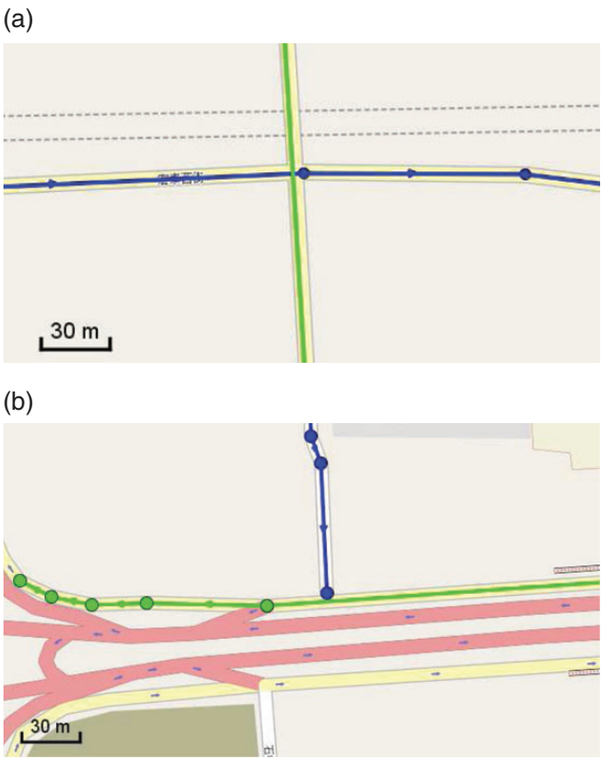

FIGURE 14. Example missing intersections in Beijing. (**a**) Touching but not connected. (**b**) Close but not connected.

Most of these incorrectly inferred intersections were due to noise. Figure 14 shows examples of the two types of missing intersections detected by our algorithm. In Fig. 14a, the node of the blue-highlighted road missed the green-highlighted road slightly. In Fig. 14b, the end node of the blue-highlighted road is <5 m from the green-highlighted road. Both are likely human mistakes. The rendered raster map (yellow background) does not reveal the disconnection at all in the first case. It does so barely in the second case if zoomed in at the maximum level. Even with highlighted nodes of road polylines, detecting both kinds of missing intersection in a city scale map would be a daunting task.

FIGURE 15. Turn restriction on bypass?

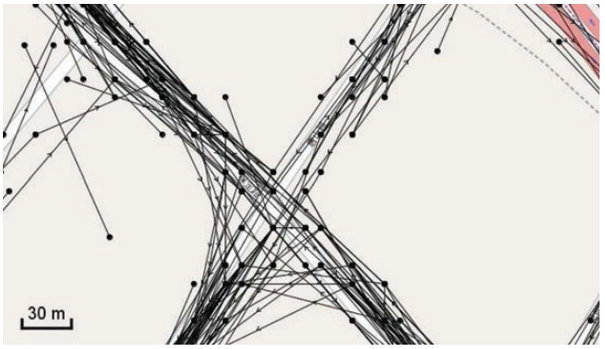

FIGURE 16. Example turn restriction in Shanghai, China.

4.2. Turn restriction detection

Using the full 21 months of data in Shanghai, there are 103 turns that do not have any matched traces at all but their incoming and outgoing segments both have at least 30 matched traces. Among these 103 turns, 70 are indeed turn restrictions not annotated by OSM. For the remaining 33 turns, 17 of them are false positives due to either insufficient data coverage or possible map changes since the GPS data was taken (we did not have the map of 2007 available, so we verified using the latest Baidu online map). The last 16 turns are on bypasses or detours. Figure 15 is an example. The blue and green-highlighted roads connect the inner circle and the outer ramp (highlighted in red). There are traces matched to both blue and green roads but not to the turn from blue to green—they took the red ramp instead. It is unclear whether the turn on the bypass is forbidden or not; however, the decision is moot, since no sensible navigation software would direct someone to take both the blue and green roads. Baidu and Google maps use a different representation for this highway interchange. Our turn restriction algorithm can ignore this type of turns by detecting whether there is a shortcut. If we set the *turnRatio* threshold to be 10% instead of 0, there are 81 turns detected and 26 of them are false positives. Figure 16 is an example turn restriction detected by our algorithm. Vehicles cannot make turns from the upper right road to the orthogonal road in either direction, or vice versa.

4.3. Road closure detection

As discussed in Section 3.5, we also apply our algorithm to the traffic data of closure segments to detect road closures, and

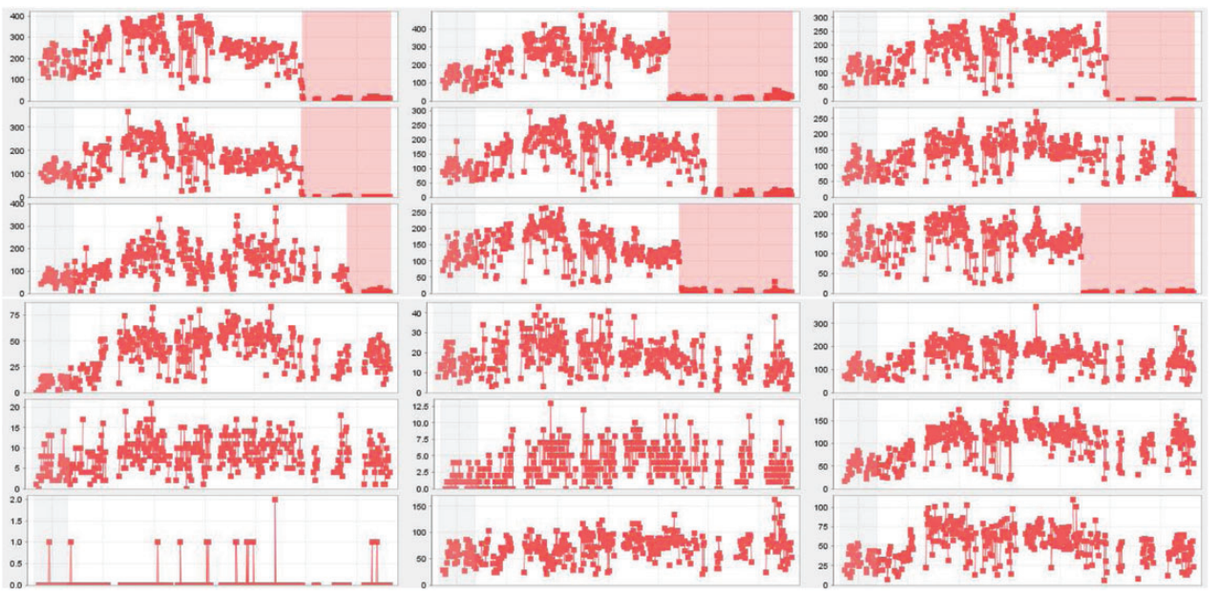

FIGURE 17. Nine example closures (top) and non-closures (bottom), according to the model. Each plot reports the number of traces per day vs. time from 03/2006 to 09/2006. All but the last of these nine closures we were able to verify by news sources or aerial imagery.

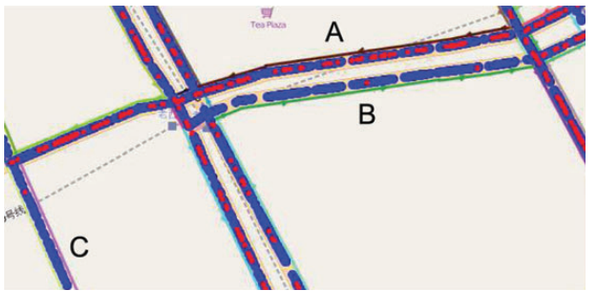

FIGURE 18. Traffic the day before (blue/lighter dots) and after (red/darker dots) detected closures at *B* and *C*.

use Google Earth historical aerial imagery and news archives to verify the results. Traffic patterns of closure segments and turns are very similar.

When we applied our algorithm to the entire dataset (5 months of data), it found 88 closures in the end that satisfied the 2-week minimum (out of 57 124 closure segments, omitting 14 000 having fewer than 10 events in the whole dataset). The identified closures ranged from 14 to 600 days, averaging 230. We attempted to verify each of these identified closures using the time and road name. In 69 of the instances we found specific news confirming the closure; in 11 further cases we verified the closure using historic aerial imagery in Google Earth. The remaining eight are unverified closures (9% of the total). Though some remain unverified, in our judgment, our algorithm achieves good precision (however, with unknown recall). Figure 17 shows the time series of trace counts for several of the identified closures and non-closures. The map view in Fig. 18 shows the traffic 1 day before and after a verified construction closure, which persisted 120 days to the end of our dataset. The blue dots show the traffic before the closure and the overlaid red dots show traffic after the closure. For closure segment *A*, there continues to be red traffic, but not for segments *B* or *C*.

5. DISCUSSION

There has been a significant amount of work on road map inference using GPS data [26], most of which depends on obtaining frequent samples from high accuracy GPS devices in one or a few probe vehicles. However, much larger volumes of GPS data are becoming available as (free) byproducts of other applications, such as fleet management and smartphones equipped with GPS receivers. These data have much lower sampling frequencies and often have less accuracy. There has been only limited recent work attempting to build road maps from such data [12, 13, 27]. These methods attempt to build a map from *tabula rasa*, which, while interesting and challenging, would present a monstrous job to try to reconcile the differences with an existing map. The map is more than just a planar graph. Road metadata aside, we find it particularly challenging to decide how many parallel polylines to use to represent a road: one for residential roads, two for arterials, four or even more for local/express freeways and parallel ramps. Thus, we believe that map *update* is a more approachable and useful area to work in. Our prior work proposed a map update framework called CrowdAtlas and focused on road updates, including geometry refinement and new road detection. There has been some published work on road geometry refinement [28–30] and map augmentation [31, 32] using high-quality GPS data. The latter focuses on using only a single trace and cannot handle noise. Intersection detection without a base map has been proposed in the past [20]. The intersection update method proposed in this paper complements our prior work, and can be easily integrated into CrowdAtlas.

Robust and accurate map matching is the cornerstone of our solution. We have categorized and compared various map matching algorithms in our prior work and have concluded that HMM-based map matching achieves a good balance between accuracy and computation speed [22]. Our algorithm and another HMM-based map matching algorithm [23] (both are variants of [16]) are the top two accurate programs in the map matching competition last year [17].

With respect to detecting openings/closures, the closest work is in traffic modeling and prediction, which attempts to identify congested traffic conditions to advise motorists. TomTom and Google both use smartphone data to estimate traffic conditions, while some researchers have tried using large-scale taxi data [33]. However, if a road is actually closed, such applications currently report 'no data' rather than a suspected closure. The probabilistic modeling work by Ihler *et al.* [34] sought traffic anomalies based on loop detector data, where the anomalies consisted of increased traffic, e.g. due to sports events. These data are very regular and periodic compared with the data we face. Later when they dealt with large-scale data, including its concomitant anomalies, they spent much of their modeling effort to detect periods where faulty loop detectors continuously reported zero traffic [35]. Although we see substantial non-zero counts even during closures, their methods could perhaps be adapted to road closure detection as well—potential future work. Our time series are quite a bit more erratic, so it would be interesting to know whether their models would work in this noisier domain.

6. CONCLUSION

Our experience with CrowdAtlas road inference and road closure detection led us to discover the substantial problems with missing intersections and turn restrictions in existing road maps. Such errors are not only frequent, but also very hard to verify with visual inspection of the map, and certainly a daunting task to verify all intersections for a large city. Lack of intersection connectivity can result in poor route planning and

confused GPS navigation aids; and missing turn restrictions can advise drivers to take illegal and dangerous forbidden turns. This prompted us to research effective methods for the automatic insertion of missing intersections and turn restrictions based on the large-scale GPS data, which are becoming increasingly available as data byproducts of other applications and processes.

We have shown the feasibility of a prototype data mining application that detects missing intersections, turn restrictions and road closures, in order to automate the updating of the digital map, which has become a cornerstone resource for so many applications. While in our research we used historic data from a fleet of over 10 000 taxis, we envision future streaming implementations that scavenge the data byproducts of a variety of fleet management deployments as well as mobile smartphones, given sufficiently anonymized, delayed and/or opt-in data sources. With ample volume and variety of available GPS streams, changes to the map can be detected and corrected in ever shorter time scales than demonstrated here. Rather than detecting a closure in 2 weeks, future systems should be able to use these data mining techniques to detect changes in real-time: minutes for busy roads and perhaps hours or days for seldom used roads. By having the most current maps, navigation aids can avoid directing drivers on circuitous routes and can avoid directing drivers to make illegal turns.

The promise shown in this paper opens the doors to multiple fronts of future work. With regard to new road detection, there is an opportunity to develop methods for dealing with noisy data in high data density regions, such as near freeways. With regard to road closure detection, it would be desirable to apply *active learning* from the beginning, in order to select the most useful examples to label. Because positives are so rare in this type of data, some kind of easy query mechanism is called for to bias the sampling toward likely positives. Additionally, it may be possible to leverage the ample unlabeled time series: either by identifying clusters of common motifs, or by semi-supervised classification of the time series [36].

While overall the results look promising for this emerging map-update application, there are naturally some limitations on the information available from vehicular GPS samples that come as byproducts—as opposed to traditional map surveys, which use frequent sampling from high grade GPS devices on planned routes. With any vehicle fleet, there will be biased sampling of the roads. Refer to Fig. 19 that shows the distribution of road types according to the OSM map, and how this compares with the distribution of GPS samples by our dataset (sorted). Observe that taxis, not surprisingly, prefer highways and major arterials, leaving the great proportion of residential roads with little coverage. Alternately, with a fleet of mail delivery carriers, one might expect the reverse. This puts a limitation on the speed with which certain road types can be expected to be detected (as either new or closed). Ideally, the application should be run with data byproducts from a variety of different fleets and other data sources.

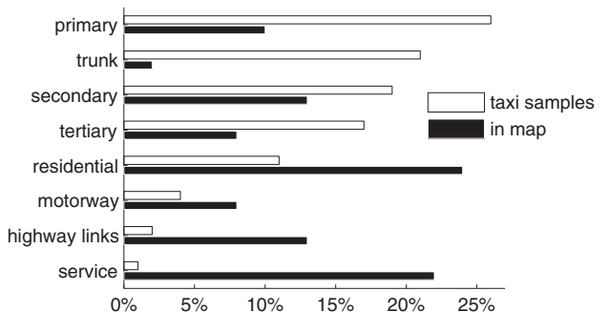

FIGURE 19. The distribution of taxi GPS samples naturally differs from the distribution of road types in the base map.

Even if a variety of complementary data sources can be gathered, there is another important limitation. With samples being taken every N seconds, one will get a preponderance of samples on slower roads. This poses a problem to obtain samples for highway onramps and offramps, which are short as well as high-speed. Thus, we should expect some types of roads will require long time horizons to accumulate enough evidence. But even in the case of a new highway onramp that receives almost no GPS samples because it is very short, if there is much traffic, then the application can automatically detect the need for a new road, though it may have to ask an editor to establish the precise geometry of the road.

FUNDING

This research was supported by 863 Program (2013AA-01A601), 973 Program (2014CB340303), NSFC (No. 614-72254, 61170238, 61420106010 and 61373157), the Science and Technology Commission of Shanghai (Grant No.14511107500) and Singapore NRF (CREATE E2S2). This work is also supported by the Program for Changjiang Scholars and Innovative Research Team in University (IRT1158, PCSIRT), China.

REFERENCES

[1] Wabash, R. (2012) *9 Car accidents caused by Google Maps & GPS*. http://www.ranker.com/list/9-car-accidents-caused-by-google-maps-and-gps/robert-wabash (accessed July 1, 2013).

[2] Vanderbilt, T. (2010) *It wasn't me, officer! It was my GPS*. http://www.slate.com/articles/life/transport/2010/06/it_wasnt_me_officer_it_was_my_gps.html (accessed July 1, 2013).

[3] Johnson, C., Shea, C. and Holloway, C. (2008) The Role of Trust and Interaction in GPS Related Accidents: A Human Factors Safety Assessment of the Global Positioning System (GPS). In Simmons, R., Mohan, D. and Mullane, M. (eds), *Proc. 26th Int. Conf. on Systems Safety, Vancouver, Canada*, Unionville, VA, USA. International Systems Safety Society.

[4] Welch, W. (2009) *More than 50 crashes on Bay bridge curve*. http://www.usatoday.com (accessed July 1, 2013).

[5] Mirror, N. (2008) *SatNav danger revealed: navigation device blamed for causing 300,000 crashes*. http://www.mirror.co.uk/news/uk-news/ (accessed July 1, 2013).

[6] Taylor, R. (2012) *Apple Australia map glitch: Snakes! In the desert!* http://www.reuters.com/article/2012/12/10/us-australia-iphone-idUSBRE8B91AQ20121210 (accessed July 1, 2013).

[7] Lomas, N. (2008) *GPS driving British motorists to distraction*. http://www.businessweek.com (accessed July 1, 2013).

[8] Kaiser, T. (2011) *GPS units lead Washington drivers into dangerous territory*. http://www.dailytech.com/GPS+Units+Lead+Washington+Drivers+into+Dangerous+Territory+/article-21947.htm (accessed July 1, 2013).

[9] Hu, J., Razdan, A., Femiani, J. C., Cui, M. and Wonka, P. (2007) Road network extraction and intersection detection from aerial images by tracking road footprints. *IEEE Trans. Geosci. Remote Sens.*, **45**, 4144–4157.

[10] Seo, Y.-W., Urmson, C. and Wettergreen, D. (2012) Exploiting Publicly Available Cartographic Resources for Aerial Image Analysis. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 109–118. ACM.

[11] Cao, L. and Krumm, J. (2009) From GPS Traces to a Routable Road Map. *Proc. 17th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 3–12. ACM.

[12] Liu, X., Biagioni, J., Eriksson, J., Wang, Y., Forman, G. and Zhu, Y. (2012) Mining Large-Scale, Sparse GPS Traces for Map Inference: Comparison of Approaches. *Proc. 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 669–677. ACM.

[13] Biagioni, J. and Eriksson, J. (2012) Map Inference in the Face of Noise and Disparity. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 79–88. ACM.

[14] Wang, Y., Liu, X., Wei, H., Forman, G., Chen, C. and Zhu, Y. (2013) Crowdatlas: Self-Updating Maps for Cloud and Personal Use. *Proc. 11th Annual Int. Conf. on Mobile Systems, Applications, and Services*, pp. 27–40. ACM.

[15] Madrigal, A. C. (2012) How Google builds its maps. http://theatlantic.com (accessed July 1, 2013).

[16] Newson, P. and Krumm, J. (2009) Hidden Markov Map Matching through Noise and Sparseness. *Proc. 17th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 336–343. ACM.

[17] Ali, M., Krumm, J., Rautman, T. and Teredesai, A. (2012) ACM SIGSPATIAL GIS Cup 2012. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems, SIGSPATIAL'12*, New York, NY, USA, pp. 597–600. ACM.

[18] OpenStreetMap. http://www.openstreetmap.org (accessed July 1, 2013).

[19] Zhu, Y. (2014) *GPS dataset*. http://grid.sjtu.edu.cn/mapmatching/data (accessed July 1, 2013).

[20] Fathi, A. and Krumm, J. (2010) Detecting Road Intersections from GPS Traces. *Geographic Information Science*, pp. 56–69. Springer.

[21] SUVnet-Trace data. http://wirelesslab.sjtu.edu.cn (accessed April 20, 2015).

[22] Wei, H., Wang, Y., Forman, G., Zhu, Y. and Guan, H. (2012) Fast Viterbi Map Matching with Tunable Weight Functions. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 613–616. ACM.

[23] Song, R., Lu, W., Sun, W., Huang, Y. and Chen, C. (2012) Quick Map Matching Using Multi-Core CPUs. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 605–608. ACM.

[24] Wang, Y., Zhu, Y., He, Z., Yue, Y. and Li, Q. (2011) Challenges and Opportunities in Exploiting Large-Scale GPS Probe Data. Technical Report HPL-2011-109. HP Labs.

[25] Liu, K., Yamamoto, T. and Morikawa, T. (2009) Feasibility of using taxi dispatch system as probes for collecting traffic information. *J. Intell. Trans. Syst.: Technol. Plan. Oper.*, **13**, 16–27.

[26] Biagioni, J. and Eriksson, J. (2012) Inferring road maps from global positioning system traces: survey and comparative evaluation. *Transp. Res. Rec.*, **1**, 61–71.

[27] Niu, Z., Li, S. and Pousaeid, N. (2011) Road Extraction using Smart Phones GPS. *Proc. 2nd Int. Conf. on Computing for Geospatial Research & Applications*, p. 22. ACM.

[28] Rogers, S., Langley, P. and Wilson, C. (1999) Mining GPS Data to Augment Road Models. *Proc. 5th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 104–113. ACM.

[29] Schrödl, S., Wagstaff, K., Rogers, S., Langley, P. and Wilson, C. (2004) Mining GPS traces for map refinement. *Data Min. Knowl. Discv.*, **9**, 59–87.

[30] Zhang, L., Thiemann, F. and Sester, M. (2010) Integration of GPS Traces with Road Map. *Proc. 2nd Int. Workshop on Computational Transportation Science*, pp. 17–22. ACM.

[31] Haunert, J.-H. and Budig, B. (2012) An Algorithm for Map Matching given Incomplete Road Data. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 510–513. ACM.

[32] Torre, F., Pitchford, D., Brown, P. and Terveen, L. (2012) Matching GPS Traces to (Possibly) Incomplete Map Data: Bridging Map Building and Map matching. *Proc. 20th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pp. 546–549. ACM.

[33] Castro, P. S., Zhang, D. and Li, S. (2012) Urban Traffic Modelling and Prediction using Large-Scale Taxi GPS Traces. *Pervasive Computing*, pp. 57–72. Springer.

[34] Ihler, A., Hutchins, J. and Smyth, P. (2006) Adaptive Event Detection with Time-Varying Poisson Processes. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 207–216. ACM.

[35] Hutchins, J., Ihler, A. and Smyth, P. (2010) Probabilistic Analysis of a Large-Scale Urban Traffic Sensor Data Set. *Knowledge Discovery from Sensor Data*, pp. 94–114. Springer.

[36] Wei, L. and Keogh, E. (2006) Semi-Supervised Time Series Classification. *Proc. 12th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 748–753. ACM.