Singapore Management University

Institutional Knowledge at Singapore Management University

Inferring accurate bus trajectories from noisy estimated arrival time records

Lakmal MEEGAHAPOLA
Singapore Management University, lakmalm@smu.edu.sg

Noel ATHAIDE
InfoSys Ltd

Kasthuri JAYARAJAH
Singapore Management University, kasthurij@smu.edu.sg

Shili XIANG
Institute for InfoComm Research

Archan MISRA
Singapore Management University, archanm@smu.edu.sg

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

Part of the Numerical Analysis and Scientific Computing Commons, Software Engineering Commons, and the Transportation Commons

Inferring Accurate Bus Trajectories from Noisy Estimated Arrival Time Records

Lakmal Meegahapola[1,a]   Noel Athaide[2,a]   Kasthuri Jayarajah[3]   Shili Xiang[4]   Archan Misra[3]

Abstract— Urban commuting data has long been a vital source of understanding population mobility behaviour and has been widely adopted for various applications such as transport infrastructure planning and urban anomaly detection. While individual-specific transaction records (such as smart card (tap-in, tap-out) data or taxi trip records) hold a wealth of information, these are often *private* data available only to the service provider (e.g., taxicab operator). In this work, we explore the utility in harnessing *publicly available*, albeit noisy, transportation datasets, such as noisy "Estimated Time of Arrival" (ETA) records (commonly available to commuters through transit Apps or electronic signages). We first propose a framework to extract accurate individual bus trajectories from such ETA records, and present results from both a primary city (Singapore) and a secondary city (London) to validate the techniques. Finally, we quantify the upper bound on the spatiotemporal resolution, of the reconstructed trajectory outputs, achieved by our proposed technique.

I. INTRODUCTION

In recent years, there has been an increased interest in the use of individualised traces of mobility data for both transportation-related analytics and broader understanding of urban human behaviour–such data includes those obtained via GPS/WiFi sensing [23], records of taxi ridership [4], [10] or bike trips [22], or consumer interactions with public transport [21], [15]). While smart card-based consumer transaction data can offer fine-grained insights (e.g., in detecting city-scale events [9]), the reality is that access to such data is very difficult to obtain (often, due to privacy concerns). In many cases, such as (origin, destination) records of taxi trips or transaction data of ride sharing services, such data are in private hands and not universally available. Moreover, even for data based on public, governmental services, public authorities usually release such data with a significant delay. This implies that such individualised data may be useful for offline, longer-term policy or operational insights, but not always usable for applications requiring soft real-time responsiveness.

To overcome both these limitations, in this work, we exploit publicly available and *aggregated* bus data, available as noisy Estimated Time of Arrivals (ETA), to tackle the problem of *accurately inferring the transit times of individual buses at bus stops in soft-real time*. The ETA estimates, which contain no information about individual-specific trips but often include aggregate occupancy information, are often made available continuously via 'live' public portals and well-document APIs, enabling external organisations to incorporate such data in third-party services. We believe that such data, when subject to intelligent processing, can be a powerful information source for studying urban mobility.

We illustrate a scenario to motivate the use of such individual bus trajectory tracing: accurate estimates of the times at which a bus stops at different bus stops allows us to more precisely extract the occupancy level changes *at* such stops. With longitudinal

observations of such fine-grained (i.e., bus stop level) occupancy (or *loading*) level changes, it is then possible to (a) build models for *normal* occupancy level changes, accounting for factors such as day of the week, time of the day, locality, etc., and (b) subsequently, use observed deviations from such normal patterns to not just detect anomalous events (e.g., [9], [10]) but even anticipate or predict anomalies in advance (e.g., see [15]).

To enable such diverse applications, we present and evaluate algorithms to accurately reconstruct the travel trajectory of vehicles from such publicly available ETA data. We make the following *Key Contributions*:

1) We propose an algorithmic framework to extract trajectories of individual buses from noisy ETA records even when the records do not contain individual bus-level identifiers. More specifically, the records periodically provide snapshot estimates of the ETA of the next K (K being a parameter) buses on a particular route, along with (optionally) the instantaneous GPS-based locations of the corresponding buses. The proposed approach first uses de-duplication to create a trajectory of a single bus instance across multiple snapshots, and then interpolates the trajectory data to infer the transit times at missing bus stops. Using a real-world dataset from Singapore, we evaluate the accuracy of the algorithms extensively and show that we can detect the *exact* time a bus transits a downstream bus stop with more than 50% precision/recall. More importantly, the precision/recall values exceed 80%, when we allow an estimation error of $\leq$ 1 minute. We also show that the *interpolation* step is crucial, as it decreases the error in estimating the arrival time of buses at downstream bus stops, by as much as $\approx$ 30% for select bus services in Singapore. We also analyse how various other factors, such as (a) the day of the week (weekdays vs. weekend) and (b) the type of route (whether the bus goes through the city centre), affect the estimation accuracy.

2) By extending our analyses to data curated from a different city (i.e., London), we show that the performance is comparable (e.g., 40-50% precision/recall in detecting arrivals with zero temporal error) and establish the generalisability of our technique. However, we also point out key differences: for example, unlike Singapore, specifying an estimation tolerance error of ±1 minute does not significantly improve the precision/recall results for London.

3) Finally, we evaluate the impact of several practical choices on the achievable accuracy. First, by varying the frequency with which the ETA updates are generated, we show that more frequent updates can result in a significant (at least 20% improvement in bus arrival time estimation accuracy). We additionally translate the impact of margins of time errors to practical upper bounds on the spatial resolution for different *target* accuracy of bus arrival detection values; we show that for both Singapore and London, a bus arrival time detection accuracy of $\approx$ 80% can be reached with a nominal spatial error (i.e., distance from the bus stop) of 200-600 meters.

[1]Idiap Research Institute & École polytechnique fédérale de Lausanne (EPFL), Switzerland. email: lakmal.meegahapola@epfl.ch

[2]InfoSys Limited, India

[3]Singapore Management University, Singapore

[4]Institute for Infocomm Research, Singapore

[a]Work done while being a Research Engineer at Singapore Management University, Singapore

II. OVERALL METHODOLOGY

We consider a dataset D consisting of tuples: $<$ $date, t, busstopID, service, eta, latitude, longitude, K$ $>$. Here, $date$ and t corresponds to the calendar day and the discretized time interval the observation belongs to, $busstopID$ and $service$ identify a unique service route and a bus stop on that route, the eta denotes the estimated time of arrival (in minutes) of the next K^{th} bus, and the (optional) $latitude, longitude$ pair captures the bus' current instantaneous location. We compute the $distance$ a bus has traveled since the starting stop based on the bus' current location.

Problem Statement: Given a sequence of bus stops, $S_n, n \in [1, N]$, for a given $service$ with N stops, and D, find m unique buses over the course of the day and their respective trajectories, T_m, which is a sequence of tuples $(m, n, t_{arrival})$ where $t_{arrival}$ is the time at which a bus m arrives at stop S_n along route $service$.

To this end, we describe our framework that consumes raw ETA observations to extract out unique trajectories of individual buses, as follows:

- **Step 1: De-duplication** – In this step, we consider pairs of consecutive bus stops to remove duplicate observations of the same bus and assign *local* identities within the same time instance (see Section II-A).
- **Step 2: Stitching** – Next, we consider such unique bus instances across consecutive snapshots in time, to extract the *trajectory*, T_m, of that instance (see Section II-B).
- **Step 3: Interpolation** – Finally, we perform interpolation to *fill* the resulting trajectory with missed (due to removal of duplicates from Step 1) transits (see Section II-C).

A. Snapshot Deduplication

We define a **snapshot**, $SS_{i,service} \subset D$, as the collection of all such tuples at $t = i$ for a representative $service$. Figure 1a illustrates the $distance$ (top) and eta (bottom) of multiple, consecutive *snapshots* of ETA observations for a representative service across all its bus stops (on the $x-$axis). The illustration shows instances of the same physical K^{th} bus across multiple consecutive bus stops as those instances where a bus has *traveled* the same $distance$ since the start of the journey (points marked in the same color in the illustration the same physical bus) within each snapshot of time. In the deduplication stage, we eliminate such duplicate instances of the same bus, as described in Algorithm 1, and illustrated in Figure 1b.

In the deduplication stage, we eliminate such duplicate instances of the same bus, as described in Algorithm 1, and illustrated in Figure 1b. We limit ourselves to $K = 3$ as the datasets used in this work (see Section III) only provide information as advanced as the subsequent 3 buses. For two consecutive bus stops (S_n and S_{n+1}), we consider the distance vectors of incoming $K = 3$ buses, X_n and X_{n+1}, respectively. We then find the optimal alignment, or 1-to-1 mapping, between the distance vectors. In this work, we resort to a simple Euclidean distance-based cost computation (lines 9 to 16) due to the number of observations/points being limited to $K = 3$. This results in three cases: (1) the three buses seen by downstream bus stop S_{n+1}, are the same seen by S_n during the same snapshot (lines 9 to 10), (2) S_{n+1}'s $K = 1$ bus has already *crossed* S_n (lines 11 to 13), (3) S_{n+1}'s $K = 1, 2$ buses have already *crossed* S_n (lines 14 to 16). After bus IDs are assigned *locally*, between pairs of consecutive bus stops, the duplicate instances of a single bus are *discarded* from the downstream bus stop. We refer to this as, dist, the distance-based deduplication technique.

In Section IV, we compare this against ETA-based de-duplication where the distance vectors are replaced by ETA-vectors. We consider an additional baseline, parametric-distance based de-duplication, which we refer to as pdist; here, we simply group together observations that are within ϵ distance of each other. As, too small a value of ϵ may result in duplicate observations being treated as distinct, and too large a value of ϵ will result in distinct observations being treated as duplicates, we resort to finding an optimal value empirically. We varied the threshold between 0 to 4000 meters, and chose the ϵ that resulted in the lowest number of snapshots that failed a success criteria. In our experiments, we set this criteria to be an ETA between two consecutive bus stops to be no more than 10 minutes, across all bus stops. In contrast to dist and eta, pdist works in an *offline* fashion as it relies on empirically learning a distance parameter that requires complete observations of a bus' journey between its terminal stops. As the applications considered in this work are geared towards *online*, soft real-time response times, dist and eta based de-duplication are more appropriate, but are evaluated against pdist for accuracy comparisons.

Algorithm 1 Local Bus ID Assignment for Stop S_{n+1} given bus IDs for observations at stop S_n

---
1: **Inputs:**
2: $X_n \leftarrow$ vector of bus distances observed at stop S_n sorted desc.
3: $X_{n+1} \leftarrow$ vector of bus distances observed at stop S_{n+1} sorted desc.
4: $I_n \leftarrow$ vector of IDs for buses observed at stop S_n
5: **Output:** $I_{n+1} \leftarrow$ **vector of IDs for buses observed at stop** S_{n+1}
6: $cost\_rr\_gg\_bb = ((X_n[1] - X_{n+1}[1]) + (X_n[2] - X_{n+1}[2]) + (X_n[3] - X_{n+1}[3]))/3$
7: $cost\_rg\_gb = ((X_n[1] - X_{n+1}[2]) + (X_n[2] - X_{n+1}[3]))/2$
8: $cost\_rb = (X_n[1] - X_{n+1}[3])$
9: **if** $cost\_rr\_gg\_bb$ is least **then**
10:      $I_{n+1} = I_n$
11: **else if** $cost\_rg\_gb$ is least **then** ▷ Generate/Increment new ID
12:      $busID = I_n[maxLength] + 1$ ▷ Append new bus to stop S_{n+1}
13:      $I_{n+1} = append(I_n, busID)$
14: **else if** $cost\_rg\_gb$ is least **then** ▷ Generate/Increment new IDs
15:      $busIDs = I_n[maxLength] + 1, I_n[maxLength] + 2$ ▷ Append new buses to stop S_{n+1}
16:      $I_{n+1} = append(I_n, busIDs)$
17: **end if**

B. Trajectory Stitching

In the previous step, we match buses across pairs of bus stops within the same time snapshot. In this step, we take the resulting bus ID assignments, and match bus IDs across time, i.e., *snapshots*, to complete the traces of individual buses, as outlined in Algorithm 2. Figure 1c illustrates the *stitched trajectory* of a single bus (spanning multiple bus stops along its route and multiple snapshots).

Similar to the previous step, we rely on simple distance-based measurements for matching, and introduce new buses where a suitable match is not found (lines 6 to 16). We illustrate the workings of Lines 9 and 10 in Figure 2 for clarity. Additionally, we introduce a speed criteria as a means for sanity check (lines 17 to 23); the speed of each bus is computed given a successful
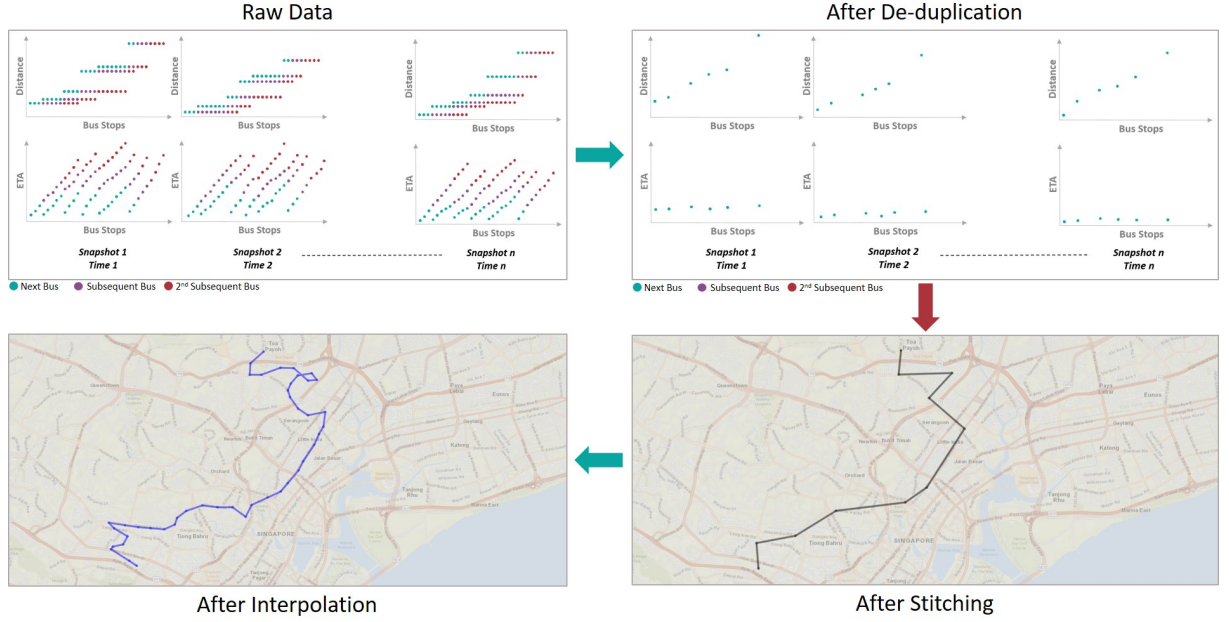
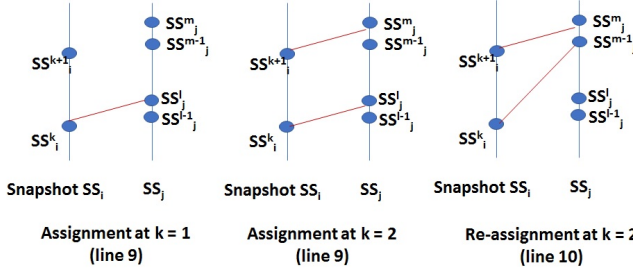Fig. 1: Overview of the algorithms described in this work.

Fig. 2: Illustration of Lines 9 and 10 of Algorithm 2.

Algorithm 2 Assign bus IDs to observations in snapshot SS_j given bus IDs for observations in snapshot SS_i

---

1: $i = 1$
2: $j = 2$
3: **while** j <= no. of snapshots **do**
4:     $SS_i \leftarrow i^{th}$ snapshot
5:     $SS_j \leftarrow j^{th}$ snapshot
6:     **for** $k = 1$ to no. of observations in SS_i **do**
7:         $C \leftarrow$ set of observations from SS_j between (SS_i^k, SS_i^{k+1})
8:         **if** C is not empty **then**
9:             $l \leftarrow argmax_{l \in |C|} dist(SS_i^k, SS_j^l)$
10:             move up RHS of all previous matchings by $|C| - 1$
11:             if any unmatched observations left from 1 to $l - 1$ in SS_j, assign new bus IDs to them
12:         **else**
13:             match SS_i^k to nearest neighbor in SS_j from SS_i^{k+1} which has travelled a greater distance
14:         **end if**
15:     **end for**
16:     **if** above matching violates speed check **then**
17:         clear matching and $j \leftarrow j + 1$ ▷ skip to next snapshot
18:     **else**
19:         $i \leftarrow j, j \leftarrow i + 1$ ▷ Save matching
20:
21:     **end if**
22: **end while**

---

match, and if the speed of any bus is greater than some threshold (in our case, we set it to 40kmph four our experiments) then we clear that matching, skip the current snapshot and repeat the matching process. The astute reader will note that due to the removal of duplicate bus instances from the previous step, the trajectory output here from this step will be a proper subset $S'_n \subset S_n$.

C. Interpolation

In the final step, we fill in for *missed* tuples (corresponding to missed bus stops due to the de-duplication step from before) in T_m. To this end, for each missed stop (S_{missed}) along the route, we consider the (1) estimated time of arrival $t_{arrival}^{missed-1}$ at the previous stop ($S_{missed-1}$) in the current trace, (2) the distance between the pair of stops, d, and (3) the historical average velocity of any bus along that service, v, and estimate the time of arrival, $t_{arrival}^{missed}$ as $t_{arrival}^{missed-1} + \frac{d}{v}$. In Figure 1d, we observe that the interpolate trajectory now consists of a more accurate representation of the bus (transiting through every stop along its route) in comparison to the stitched trajectory from before. Now, $|T_m| = N$.

III. DATA DESCRIPTION

We consider a longitudinal, bus dataset from Singapore, as our primary dataset in this work. We collected Estimated Arrival Time (ETA) records of incoming buses for over 401 bus services from over 4913 bus stops island-wide using the pub-

licly available DataMall API [1]. Each record is a tuple $<busstopid, serviceid, timestamp, loading, ETA, loc_x, loc_y>$ where $busstopid$ and $serviceid$ identify the bus stop and the service, respectively, and $loading$ is a discrete number varying between 1 and 3 capturing the level of crowdedness of the next bus scheduled to arrive at that stop, the ETA representing the

---

[1] https://www.mytransport.sg/content/mytransport/home/dataMall.html

estimated arrival time of the bus, and loc_x and loc_y representing the GPS coordinates in at query time. The API is refreshed every 60 seconds, with $timestamp$ capturing the most recent refresh instant. In addition to the next immediate incoming bus, the data also provides the same information of the subsequent bus and the bus thereafter, when available. We consider the locations of bus stops and the service route information [2] to convert the two-dimensional location of incoming buses to a uni-dimensional *distance* measure, i.e., the distance the bus has traversed since its origin stop.

Additionally, we collected data from London, using the Transport for London Unified API [3], to validate the methodology presented in this work. In contrast to the Singapore dataset, the presence of a field capturing the identity of the vehicle allows us to use this dataset as means for additional validation (see Section IV) albeit not consisting of the GPS coordinates of the buses' instantaneous location. The records are refreshed every 30 seconds. We summarize key statistics related to the datasets in Table I.

City	Observation Period	Services Considered	Sampling Rate
Singapore	2018-06-01 to 2018-07-19	124, 147, 190 139, 32, 65, 195	60 seconds
London	2018-07-04 to 08	12, 17, 326, 111	30 seconds

TABLE I: Summary of datasets used in the analysis.

A. Preliminaries

As we mentioned previously, our methods for extracting bus trajectories from ETA records are inherently noisier, as compared to, the often *protected* smart fare card transactions-based data due to: (1) the inability to distinguish between individual buses (as the bus IDs are usually unknown), (2) and the timing of when a bus actually crosses the individual stops along its route being unknown.

Here, we discuss some key scenarios (that we observe in our dataset) that challenge accurate trajectory discovery. To quantify the intensity of these possible problem scenarios, we rely on an additional dataset consisting of smart-fare based transactions which consist of the details of trips made by commuters, including start and end stops, start and end time of the trip, the service route and registration number of the bus, during the month of August 2013.

Paired trajectories: We observe a non-negligible number of instances (e.g., 43.96% cases over in service 190 & 22.94% cases over 5 service routes) where two or more buses servicing the same route to be closely following each other, especially during peak hours. Without knowing the identity of the individual buses, the algorithms presented in this work are likely to cluster such buses as a single bus instance.

Skipped stops: As commonly seen across cities, buses skip certain bus stops when there are no alighting or boarding passengers for an upcoming stop. We observe that on a typical weekday, on average 24.78% stops are skipped.

On-demand activation of surplus buses: Typically, buses start and end their journeys at stipulated terminus stops for that service route. In our data, we observe that, especially during peak hours, there are a number of instances (e.g., 5.29% additional buses in operation, on a typical weekday PM peak hours) where buses are activated *on-demand*, which don't necessarily start their journey from the beginning of the route (illustrated in Figure 3) which introduces errors as the algorithms take into account, the distance the bus has travelled since the start.
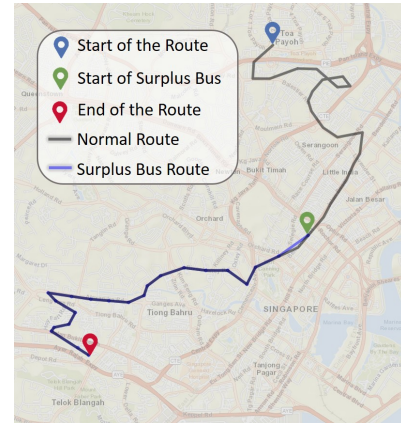
Fig. 3: Original Route and the Route of a Surplus Bus in Route 139, Direction 1

IV. EXPERIMENTS

In this section, we summarize results from our experiments in evaluating our algorithms along a number of dimensions.

A. Performance Metrics

We evaluate three main aspects of our combined framework: (1) the accuracy of detecting individual buses, (2) the accuracy with which the crossing of a bus at a particular bus stop is detected (i.e., arrival detection), in Section IV-B, and (3) the impact of practical considerations such as sampling frequency on the limits to performance in Section IV-C.

We measure the effectiveness of the algorithm in detecting individual buses by comparing the frequency of detected buses against actual transit schedules. We report the percentage of cases where the algorithm under, or, over-reports the frequency for each $busservice$.

For each distinct $< busservice, busstop >$ pair, we consider buses that cross the stop as the set of *observed* buses, and consider that the algorithm *detected* that bus, if the arrival time of a bus (of the same service for that bus stop) is within a time, Δ_t, of the actual arrival time t_{actual} of that bus. We then follow standard definitions in reporting the accuracy in terms of precision (Eq. 1) and recall (Eq. 2) where Obs is the set of all observed buses, Det is the set of all detected buses, and $|.|$ denotes the cardinality operator.

$$precision = \frac{|Obs \cap Det|}{|Det|} \quad (1)$$

$$recall = \frac{|Obs \cap Det|}{|Obs|} \quad (2)$$

B. Bus Trajectory Tracking

1) Frequency Analysis: To understand the effectiveness of our algorithms, we first study at an aggregate frequency level over multiple time windows across the day. We consult publicly available schedule information[4], to extract the approximate number of buses expected to be in service for four different times of the day: AM (6-10:29am), AMOP (10.30am-3:59pm), PM (4pm-7.59pm), and PMOP (8pm-5:59am). In Figure 4, we plot the percentage of cases where the detected number of buses by our algorithms (distance-based, ETA-based and parametric distance-based) were
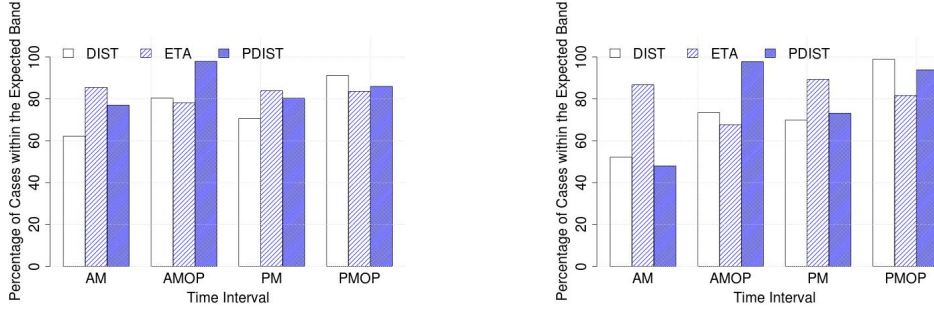
Fig. 4: Percentage of bus arrival estimation within the expected scheduled frequency, for (a) weekdays and (b) weekends, for different hours of the day.

within the band reported in the schedule information, for weekdays and weekends, separately. We observe that the ETA-based method performs consistently better than the methods (except for during AM Off-Peak hours), and that the accuracy is $\geq 80\%$ in general, except for AM Off-Peak hours during the weekend.

Next, in Table II and Table III, we summarize the accuracy of detection, for a sample set of service routes (i.e., 139, 65, 32 and 195). Except for service 195 which is *looped*, the remaining three services serve in 2 directions. The zero-error margin results correspond to cases where we consider that the detection was correct only if the number of buses detected fall exactly within the *scheduled* band, and the 10% error margin corresponds to cases where the detections are within a +/- 10% of the scheduled band. We see that the algorithms are less accurate over weekends and during off peak hours.

2) Arrival Time Detection: Here we focus on the ability of the algorithms in detecting the exact instance an individual bus crosses a downstream bus stop along its service route.

Groundtruth data: To report the accuracy of bus arrival detection, in the case of Singapore, we collected additional data where we manually observed and annotated buses crossing particular bus stops along with the time, at the minute level, at which they crossed (t_{actual}). We collected data over 7 days between 5 PM and 7 PM (i.e., evening peak hours) from two different bus stops, one within the Central Business District (CBD), and another in the outskirts of the CBD area (non-CBD). In total, we collected 28 hours of annotated bus crossings, consisting of 3173 bus crossings ($|Obs|$).

In Figure 5, we plot the precision and recall of detecting the exact time a given bus crossed a chosen bus stop (near our university campus), to the closest minute, for three different bus services, at different stages of the algorithm: de-duplication, stitching and interpolation. We find that the accuracy to be in the 40-50% range while using ETA (while using distance alone performs generally worse).

As anticipated, the accuracy after the stitching stage is poor – this is due to the fact that this stage, by design, removes multiple

Service	Zero error margin				10% error margin			
	AM	AMOP	PM	PMOP	AM	AMOP	PM	PMOP
139-1	0	20	60	40	4	68	92	92
139-2	0	57.7	96.2	88.5	0	96.2	100	96.2
32-1	72	76	96	60	100	96	96	88
32-2	3.8	7.7	96.2	73.1	88.5	61.5	100	96.2
65-1	42.3	30.8	96.2	19.2	88.5	57.7	100	84.6
65-2	34.6	30.8	96.2	57.7	92.3	65.4	96.2	96.2
195	88	8	44	12	100	44	100	72

TABLE III: Summary of accuracy in detecting individual buses, for different bus services, on **weekends** using the *ETA*-based algorithm.

(a) DIST, After De-duplication   (b) ETA, After De-duplication

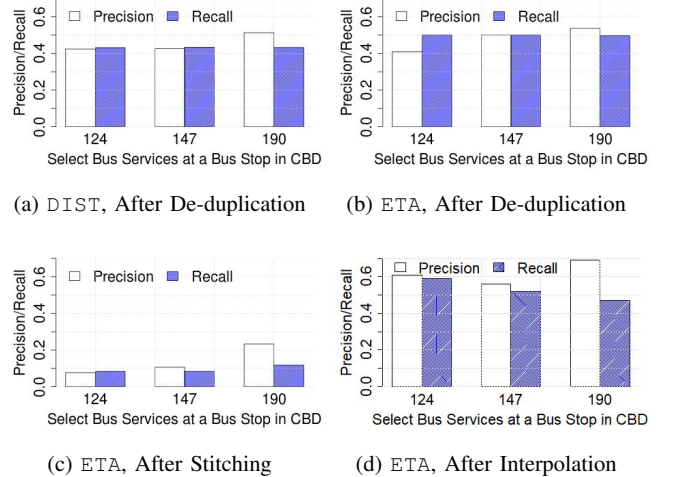(c) ETA, After Stitching   (d) ETA, After Interpolation

Fig. 5: Precision/Recall of Arrival Detection for (a) distance-based deduplication (DIST), (b) ETA-based deduplication (ETA), (c) ETA-based output after stitching, with $\Delta_t = 0$, and (d) after velocity-based interpolation, for Singapore.

occurrences of a bus's trajectory. The interpolation stage then use these sparse, albeit accurate observations, to output the final trajectory. In Figure 6, we plot the accuracy values for the case of London, and note that the performance remains comparable.

C. Practical Implications

Performance vs. Error Margin (Δ_t): Thus far, we have evaluated the effectiveness of our algorithms in distinguishing between distinct, physical buses and the ability to detect exactly when a bus crosses its downstream stops. However, we note that for several practical applications, knowing the *exact* time of arrival (i.e., a zero margin of error) may not be critical. Here, we increase the error margin systematically, and observe its impact on the performance. In Figure 7, we plot the allowed error margin on the $x-$axis (varied between 0, 1 and 2 minutes), and the precision/recall on the $y-$axis for (a) Singapore and (b) London. We observe that the performance
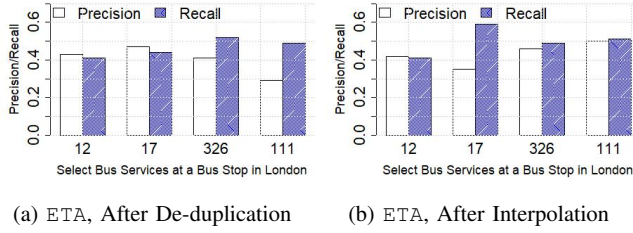
Service	Zero error margin				10% error margin			
	AM	AMOP	PM	PMOP	AM	AMOP	PM	PMOP
139-1	14.8	27.4	88.7	51.6	75.4	64.5	100	98.4
139-2	3.2	71	98.4	91.9	14.5	98.4	100	98.4
32-1	88.9	77.8	98.4	71.4	95.2	95.2	100	95.2
32-2	46.8	24.2	100	61.3	88.7	71	100	96.8
65-1	33.9	33.9	100	66.1	88.7	67.7	100	91.9
65-2	75	40.6	98.4	93.8	93.8	84.4	100	100
195	50	6.5	14.5	0	90.3	12.9	85.5	6.5

TABLE II: Summary of accuracy (in percentage) in detecting individual buses, on **weekdays** using the *ETA*-based algorithm.

(a) ETA, After De-duplication

(b) ETA, After Interpolation

Fig. 6: Precision/Recall of Arrival Detection for (a) ETA-based de-duplication (ETA), and (b) after velocity-based interpolation, for London.

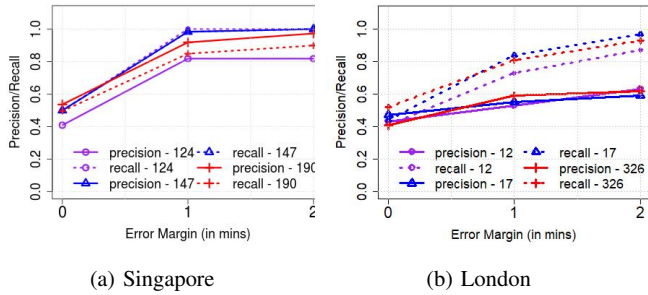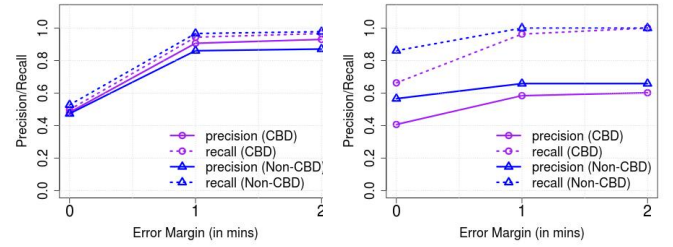(a) Singapore

(b) London

Fig. 7: Precision/Recall of Arrival Detection for varying time error margins (Δ_t) for each bus service in the CBD after deduplication for (a) Singapore and (b) London.

(a) Sampled every 2 minutes (Sin-(b) Sampled every 30 seconds (Sin-gapore)   gapore)

(c) Sampled every 2 minutes (Lon-(d) Sampled every 30 seconds (Lon-don)   don)

Fig. 8: Precision/Recall of Arrival Detection for varying time error margins (Δ_t) with bus services averaged in CBD vs. non-CBD areas, in (after de-duplication) for different sampling frequencies, in Singapore and London.

reaches its maximum with a minute's error, in both cases – in other words, our algorithms are able to detect *most* arrivals (with precision/recall ≥ 0.80) within a minute of them occurring. We also observe that allowing for larger margins (≥ 2 mins) doesn't necessarily increase the performance.

Performance vs. Sampling Frequency (F_s)**:** Next, we study the impact of the sampling frequency on the effectiveness of the algorithms. Note that thus far, we have resorted to $F_s =2$ minutes (*low*) and $F_s =30$ seconds (*high*), for Singapore and London, respectively, as were the recommended update frequencies. In Figure 8, we plot the precision/recall of arrival detections, for varying permissible error margins, for both *low* and *high* frequencies, for services running through the CBD and otherwise. We observe that by more frequent polling, the recall improves significantly (e.g., from 50% to 85% in the case of non-CBD services in Singapore, with zero temporal error). However, we also note that frequent polling results in no observable improvement with increased error margins. We also point out, that whilst we see no significant difference in performance between CBD and non-CBD areas, with more frequent polling, we see that the algorithms perform better in detecting arrival times in the non-CBD area (e.g., $\approx$ 20% improvement in both precision and recall with zero time error, over CBD buses).

Performance vs. Error in Interpolation: As we describe in Section II, the last step in our algorithm, interpolation, outputs the arrival times of buses at each stop downstream when the corresponding data points are removed during the de-duplication stage. As this might introduce errors, we next study the extent to which such approximation errors impact the overall performance. In Figure 9, we plot the precision/recall for services running in CBD and non-CBD, for zero error margin, in (a) Singapore and (b) London, for *low* (i.e., 1239.4 m or less in Singapore, 1032.7m or less in London – these values are mean values for each city) and *high* interpolation distance error. As anticipated, we note that the
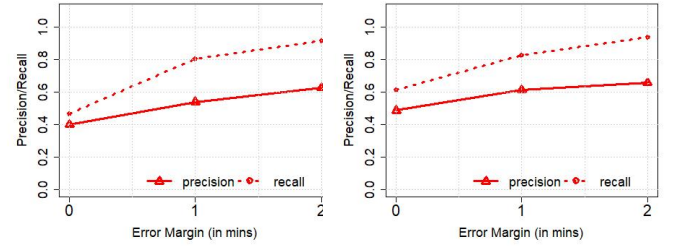
observed accuracy is lower for cases where we make a possibly larger error during interpolation (i.e., distance between the closest upstream and downstream bus stops of a stop of interest is large) – for instance, we see that in the case of Singapore, with zero error margin, we see at least a 15-20% increase in both precision and recall for cases where the interpolation error is *lower* than the mean error over all cases. For London, we see that this improvement is more pronounced (*approx* 30%).

Upper Bound on Spatial Resolution: As a result of the *temporal range* or *error margins* within which the algorithms detect arrivals, there is an inherent upper bound on the *spatial resolution* with which applications that consume such resulting bus trajectory data (coupled with occupancy levels) can operate practically. In order to understand this, we plot the achievable, or target accuracy, of a potential application, on the $x-$ axis, and the spatial resolution on the $y-$axis, after the de-duplication stage in Figure 10. We compute the spatial resolution by first extracting the time error margin (extending beyond the maximum of 2 minutes we have thus far considered, where required) that results in the corresponding target accuracy (solid lines correspond to precision, and dashed lines correspond to recall), and then multiplying the temporal error by the average speed of any bus traversing through the corresponding service route. Unsurprisingly, the spatial resolution is significantly less for buses servicing non-CBD routes (e.g., maximum achievable accuracy is reached with a resolution $\leq$ 400 m in CBD (with the exception of precision for the 147 route) whilst the same is achieved with resolution in the 700 meters range in non-CBD areas). As an aside, we also note that the maximum achievable accuracy is service route-dependent and does not monotonically increase with increased permissible error margins (but saturates beyond a point, as observed previously). Further, in Figure 11, we observe that the spatial resolution achieved in London compare similarly –
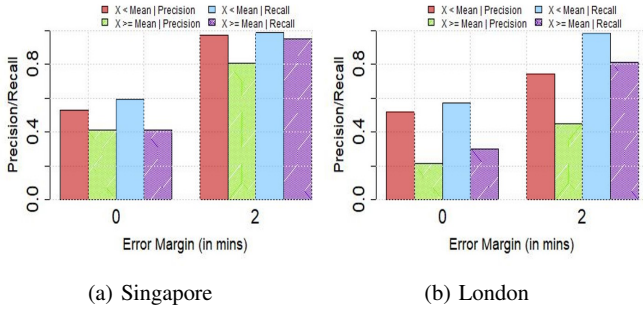
(a) Singapore   (b) London

Fig. 9: Performance stratified by distance in interpolation with san
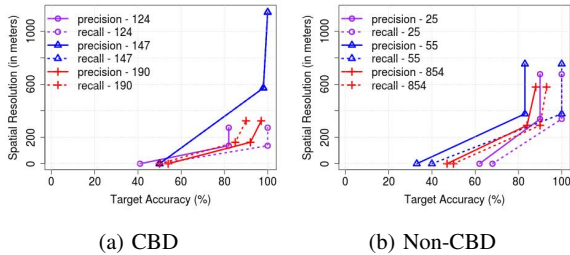

(a) CBD   (b) Non-CBD

Fig. 10: Upper bound on spatial resolution for practical applications with de-duplication, for three bus services each in the (1) CBD area and (2) n
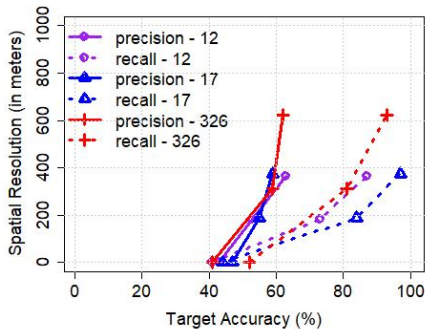

Fig. 11: Upper bound on spatial resolution for practical applications after the de-duplication phase, for three bus services each in the London dataset.

for instance, for recalls of between 80-100%, we observe that in both cases, Singapore and London, the resolution is in the 200-600 meters range.

D. An Illustrative Use Case: Bus Arrival Time Prediction

We anticipate that the ability to accurately trace the trajectory of public buses, *at real time*, using publicly available *ETA* records alone can be useful for many practical applications. We take an example application to demonstrate the utility of using real-time trajectory information of buses as opposed to relying on historic records – in particular, in Figure 12, we plot observations from *predicting* arrival times of buses for downstream bus stops.

For this analysis, we consider bus services {7,14,16,36,147,190 and 857} all of which pass through the bus stop (S_n) considered in Section IV between the PM peak period of 5 PM and 7 PM, over all weekdays. In the real-time case, we consider observations an hour prior to the prediction time, to compute the average speeds of buses along the road segment that leads to this specific bus stop (S_n) from the upstream bus stop (S_{n-1}), depending on the service route). The prediction task involves predicting the *estimated arrival*
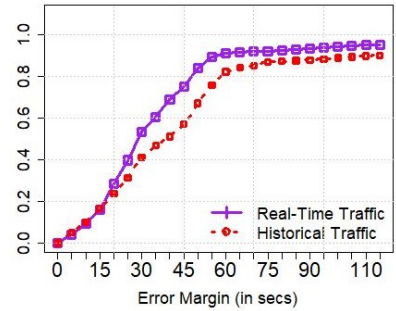

Fig. 12: CDF of temporal error margins for bus arrival time predictions with *real-time, stitched trajectories* vs. historical ETA records.

time of the outgoing buses at the next stop, during the test period of 5 PM to 7 PM on a specific day (i.e., June 6th 2018), based on the actual time of arrival at S_{n-1}, the average speed along the segment $S_{n-1} \rightarrow S_n$ and the length of the segment. The speed information here is computed on a per-bus basis due to the individual bus-level trajectory tracing made available by the technique described in the earlier sections.

In the historic case, we compute the speed along the segment based on the GPS traces of the unprocessed *next buses* over weekdays from two past months (April to May 2018). In Figure 12, the $x-$axis represents the temporal error margins over predictions for all buses during the two-hour test window whilst the $y-$axis represents the CDF for the two cases: (1) real-time, from stitched trajectories, vs. (2) historic, unstitched ETA records. We observe that while both methods are able to predict 20% of the cases with a small error margin (e.g., ≤20 seconds), the real-time method outperforms for the remaining 80% of the cases. For instance, whilst 75% of the predictions are within an error margin of 45 seconds with the real-time data, the same is true only for 55% of the cases using historic records. This observation clearly demonstrates the added benefit of being able to reconstruct trajectories at real-time.

V. RELATED WORK

Using transportation data for urban planning and analytics has a long and fruitful history (obtained via taxi ridership [4] or bike trip [22] records, public transport data [21], [15]). Recently, there have been a wealth of research that study traffic flow estimations using large datasets [16], [8], [11], [1]. While Meng et al. [16] address the problem of estimating city-wide traffic volumes using loop detectors deployed across the city and taxi trajectories, Hoang et al. [8] attempt to forecast city-wide crowd-flows based on big data. Li et al. [11] demonstrate promise in predicting travel times using only a small number of *GPS* cars. In other studies, researchers have also used transportation and mobility data to interesting and varied uses such as automatic transit map generation [18], modeling air pollution densities [13] and visualization of trajectories and crowd flows [1]. Further still, detecting anomalies in urban mobility patterns from physical sensors such as GPS traces and traffic cameras [3], [14], and CDR [19], [20] is a well-studied topic in the context of optimizing traffic related infrastructure. CDR data have also been used to detect unusual urban events (e.g., elections, emergency events, etc. [6], [7]). Previous works on anomaly detection in transportation have looked at varied aspects of the problem including the detection of and finding the root causes of the anomalies. Pang et al. [17] detect anamolous regions using Likelihood Ratio Tests and, Liu et al. [14] proposed a formulation for detecting anomalous road blocks using observed

minimum distortion and associating causality using frequent subtree mining. We highlight that many of these works rely on private data (obtained directly from transport authorities, or taxicab operators) that is available as stored datasets that allow for studying urban planning related problems as offline case studies. On the contrary, the algorithms we propose here are designed to work with *publicly available*, *real-time* data (available in many metro cities worldwide) which not only useful in offline studies and but also enable the building of practical systems with real-time response times.

There have been several published works on predicting bus arrival times [24], [5], [12], [2] – we clarify that these works focus on predicting the arrival times (or, ETA) of buses *more accurately*, and are complementary to our work in that the algorithms presented in this work consume ETAs as input to build trajectories of buses whose identities are not known.

VI. CONCLUDING REMARKS

In this work, we demonstrate the ability to trace individual bus trajectories from anonymized, *live* ETA records, available in the public domain. We show that the localization of individual buses to downstream bus stops can be performed within reasonable temporal error bounds ($\leq$ 1 min), and provide quantitative insights on how operational parameters (such as the ETA reporting frequency) affect the localization accuracy. Our evaluations from two major metropolitan cities, Singapore and London, demonstrate that the live, bus-tracking technique presented here can enable applications that were previously only possible with private, fine-grained data (such as smart-fare card transactions).

VII. ACKNOWLEDGMENT

REFERENCES

[1] A. Bessa, F. d. M. Silva, R. F. Nogueira, E. Bertini, and J. Freire, "Riobusdata: Outlier detection in bus routes of rio de janeiro," *arXiv preprint arXiv:1601.06128*, 2016.

[2] Y. Bin, Y. Zhongzhen, and Y. Baozhen, "Bus arrival time prediction using support vector machines," *Journal of Intelligent Transportation Systems*, vol. 10, no. 4, pp. 151–158, 2006.

[3] S. Chawla, Y. Zheng, and J. Hu, "Inferring the root cause in road traffic anomalies," in *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, 2012, pp. 141–150.

[4] M. F. Chiang, T. A. Hoang, and E. P. Lim, "Where are the Passengers? A Grid-based Gaussian Mixture Model for Taxi Bookings," in *ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL)*, 2015.

[5] S. I.-J. Chien, Y. Ding, and C. Wei, "Dynamic bus arrival time prediction with artificial neural networks," *Journal of Transportation Engineering*, vol. 128, no. 5, pp. 429–438, 2002.

[6] Y. Dong, F. Pinelli, Y. Gkoufas, Z. Nabi, F. Calabrese, and N. V. Chawla, "Inferring unusual crowd events from mobile phone call detail records," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2015, pp. 474–492.

[7] D. Gundogdu, O. D. Incel, A. A. Salah, and B. Lepri, "Countrywide arrhythmia: emergency event detection using mobile phone data," *EPJ Data Science*, vol. 5, no. 1, p. 25, 2016.

[8] M. X. Hoang, Y. Zheng, and A. K. Singh, "Fccf: forecasting citywide crowd flows based on big data," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2016, p. 6.

[9] K. Jayarajah, A. Misra, X.-W. Ruan, and E.-P. Lim, "Event detection: Exploiting socio-physical interactions in physical spaces," in *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE, 2015, pp. 508–513.

[10] K. Jayarajah, V. Subbaraju, N. Athaide, L. Meegahapola, A. Tan, and A. Misra, "Can multimodal sensing detect and localize transient events?" in *Proceedings Volume 10635, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IX*, ser. SPIE Defense + Security '18, 2018. [Online]. Available: https://doi.org/10.1117/12.2322858

[11] Y. Li, D. Gunopulos, C. Lu, and L. Guibas, "Urban travel time prediction using a small number of gps floating cars," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2017, p. 3.

[12] W.-H. Lin and J. Zeng, "Experimental study of real-time bus arrival time prediction with gps data," *Transportation Research Record: Journal of the Transportation Research Board*, no. 1666, pp. 101–109, 1999.

[13] Y. Lin, Y.-Y. Chiang, F. Pan, D. Stripelis, J. L. Ambite, S. P. Eckel, and R. Habre, "Mining public datasets for modeling intra-city pm2. 5 concentrations at a fine spatial resolution," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2017, p. 25.

[14] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, "Discovering spatio-temporal causal interactions in traffic data streams," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 1010–1018.

[15] L. Meegahapola, T. Kandappu, K. Jayarajah, L. Akoglu, S. Xiang, and A. Misra, "Buscope: Fusing individual & aggregated mobility behavior for "live" smart city services," in *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '19. New York, NY, USA: ACM, 2019, pp. 41–53. [Online]. Available: http://doi.acm.org/10.1145/3307334.3326091

[16] C. Meng, X. Yi, L. Su, J. Gao, and Y. Zheng, "City-wide traffic volume inference with loop detector data and taxi trajectories," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2017, p. 1.

[17] L. X. Pang, S. Chawla, W. Liu, and Y. Zheng, "On mining anomalous patterns in road traffic streams," in *Proceedings of the 7th International Conference on Advanced Data Mining and Applications - Volume Part II*, ser. ADMA'11, 2011.

[18] R. Verma, S. Ghosh, N. Ganguly, B. Mitra, and S. Chakraborty, "Smart-phone based spatio-temporal sensing for annotated transit map generation," in *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2017, p. 16.

[19] P. Widhalm, Y. Yang, M. Ulm, S. Athavale, and M. C. González, "Discovering urban activity patterns in cell phone data," *Transportation*, vol. 42, no. 4, pp. 597–623, 2015.

[20] M. Yin, M. Sheehan, S. Feygin, J.-F. Paiement, and A. Pozdnoukhov, "A generative model of urban activities from cellular data," *IEEE Transactions on Intelligent Transportation Systems*, 2017.

[21] N. J. Yuan, Y. Wang, F. Zhang, X. Xie, and G. Sun, "Reconstructing Individual Mobility from Smart Card Transactions," in *IEEE International Conference on Data Mining (ICDM)*, 2013.

[22] H. Zhang, Y. Zheng, and Y. Yu, "Detecting urban anomalies using multiple spatio-temporal data sources," *Journal of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 2018.

[23] M. Zhou, M. Ma, Y. Zhang, K. Suia, S. Pei, and T. Moscibroda, "EDUM: Classroom Education Measurements via Large-scale WiFi Networks," in *ACM International Joint Conference on Pervasive and Ubiquitous Computing (Ubicomp)*, 2016.

[24] P. Zhou, Y. Zheng, and M. Li, "How long to wait?: predicting bus arrival time with mobile phone based participatory sensing," in *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 2012, pp. 379–392.