Evaluating Interplay between Trajectory Segmentation and Mode Inference Error

Transportation Research Record
2023, Vol. XX(X) 1–19

DOI: 10.1177/ToBeAssigned
journals.sagepub.com/home/trr
SAGE

Gabriel Kosmacher[1] and K. Shankari[1]

Abstract

Travel behavior changes are essential to transportation decarbonization. Travel diaries, consisting of sequences of trips between places, are typically used to instrument human travel behavior. However, these diaries are only as accurate as the underlying methods used to construct them. Travel diary algorithms have been a popular research topic since the advent of Global Positioning System (GPS) tracking surveys. These algorithms have typically been validated using prompted recall of presegmented trips, thus disregarding the continuity of mode inference. Phone operating systems have adopted battery-conserving techniques, but the resulting data collection errors have not been studied extensively. We introduce a framework to evaluate accuracy of trip length computations and mode inference by analyzing continuous mode-segmented trajectories for groups of trips. We then use the framework to identify the input data quality and the impact of postprocessing. Our primary inputs to this evaluation are MobilityNet, a public dataset containing information from three artificial timelines covering 15 different travel modes, and sample open-source travel diary creation algorithms from the OpenPATH project. Our framework concretely shows the variance of the distance error drops from $(0.217, 0.0848)$ to $(0.011, 0.0407)$ (Android, iOS) after postprocessing. Similarly, weighted F-scores for mode inference increase from $(0.25, 0.29)$ to $(0.60, 0.74)$ (iOS, Android) between random forest and Geographic Information Systems (GIS)-based models. We hope that this standardized method will be adapted to evaluate other, potentially proprietary, travel diary algorithms. The results can be used to understand and improve the state of the art in the travel diary creation field.

Transportation is the largest source of greenhouse gas emissions in the United States (*1*). Reducing transportation emissions depends on human travel behavior, which relies on strategic local land use and planning. Travel diaries, consisting of sequences of *trips* between *places* for a particular individual, are typically used to instrument human travel behavior.

Travel diaries have historically been entirely self-reported, with initial personal visits (*2*) modernized first by telephone surveys (*3*) and then by in-vehicle Global Positioning System (GPS) surveys (*4*) and web-based surveys (*5*). The travel diaries include end-to-end information such as start and end times and a rich set of transportation modes—the 2009 NHTS included support for 25 modes, including intercity buses, paratransit and neighborhood electric vehicles (NEVs)—and semantic information such as trip purpose. More recently, general GPS-based (*6*) or smartphone app-based (*7*) travel surveys have been used to reduce respondent burden through automated trip detection and improved accuracy, particularly for trip distance (*8*) and short, nonmotorized trips (*9*).

These travel surveys have typically been used to build activity-based models (*10*, *11*) or four-step models (*12*) at the regional level to assess the impact of future changes and prioritize infrastructure improvements. Smartphone-based travel diaries have also been used for longitudinal data collection and direct evaluation of programs aimed at shifting behavior (*13*).

However, the information gleaned from travel diaries is only as accurate as the underlying methods used for data collection. While there has been much work implementing systems to instrument travel behavior, there exists no consensus procedure of evaluation. This lack of consensus is spurred by a myriad of reasons. For instance, much of the research conducted in the instrumentation of travel behavior pertains to the use of proprietary algorithms and applications (*14*, *15*), leading to the reluctance of many developers to divulge details about their research. Another hindrance in developing a consensus procedure of evaluation is that many such systems to evaluate travel behavior exist in the personal, not social, domain. In the personal domain, users make decisions based on self-tracking and

[1]National Renewable Energy Laboratory, Golden, CO

Corresponding author:
K. Shankari, k.shankari@nrel.gov

Prepared using TRR.cls *[Version: 2020/08/31 v1.00]*

have an intuition of accuracy based on experienced ground truth. The decisions are low-stakes lifestyle changes, which may have individual meaning but little social consequence. Yet when these systems are implemented in the social domain, accuracy becomes a requirement. A metropolitan transportation agency tasked with allocating millions of dollars in funding needs to know the accuracy of the data before making decisions (*16*).

Despite the lack of consensus on evaluating systems instrumenting travel behavior, travel diary algorithms have been a popular research topic since the advent of GPS tracking surveys. Thus, we look to this domain in constructing a travel diary evaluation methodology. In particular, we look to representations in literature of both geospatial point collection and mode inference algorithms. However, research in both subjects lacks information pertaining to automated travel data collection in the following ways:

Geospatial point collection: Analysis of geospatial points has focused on individual point-based error, but there has been little insight on how these errors propagate throughout a trip, affecting total trip length measurements.

Mode inference algorithms: Validation of mode inference has typically used prompted recall of presegmented trips or edge buffering to match segments, neither of which account for segmentation error, thus disregarding the continuity of mode inference.

Classical methods of analysis, which rely on making discrete point-based comparisons, provide significantly less information than evaluating travel diaries as a continuous stream of transportation mode segmented data ((*17*)). The aforementioned work proposed a time interval alignment method evaluation methodology—resulting in a slew of precision, recall, and cardinality metrics— to capture these continuous errors. We build off this work by simplifying the temporal alignment process, resulting in a concise set of metrics that are suitable for incorporation into downstream analysis, and performing a case study on a high-granularity and open-source dataset.

Compounding the issue of error analysis is that phone operating systems and applications have adopted battery-conserving techniques to limit background operation. For example, Android has introduced Doze Mode, which throttles background operation when the phone is not in use (*18*). iOS has always restricted background app operation unless actively using certain features such as location (*19*). And smartphone apps can choose to reduce accuracy or frequency of data collection when not in active use (*20, 21*). However, we are not aware of prior work that characterized the resulting data collection errors or evaluated procedures to mitigate them.

We introduce a methodology to evaluate accuracy of trip length computations and mode inference. To determine accuracy of trip length, the methodology analyzes measurement error for *minimal groups* of trips, avoiding double-counting that would otherwise be induced by faulty trip segmentation. Mode inference accuracy is determined by analysis of continuous mode-segmented trajectories using a novel temporal alignment procedure, a derivative of the alignment procedure in (*17*), itself a variation of time algebra alignment proposed by (*22*).

The remainder of this paper is organized as follows. The next section reviews related work. The following two sections provide necessary definitions and introduce the evaluation methodology. We then illustrate how the evaluation methodology can characterize the change in distance and inference error (i) for different stages in a pipeline, from the raw input to the final, postprocessed output, and (ii) across different algorithms for the same stage. Finally, we conclude and discuss future work.

Related work

Both the study of accuracy in location-based services and classification of transportation modes are popular in literature. This section reviews previous research in evaluating trip length and mode inference, focusing on work directly pertaining to the scope of this paper.

Trip length

Passive travel survey smartphone applications typically use notoriously unpredictable (*23*) *fused* location data—modern iPhones do not allow access to GPS directly (*24*)—and thus errors in these geospatial points are inherent to trip length error. Researchers have shown many factors that affect geolocation collection on smartphones, from indoor locations (*25*) to underlying APIs (*23, 26*) to how a user physically handles the phone (*27*). These errors exist in conjunction with the systematically overestimated distance by GPS as shown in (*28*).

However, none of these studies provides insight into how these errors propagate throughout a passive data collection run to affect total trip length measurements. We expand our search to errors in other passively collected travel surveys, where there has been considerably less work. (*29*) proposes a method to address missingness in geolocation data collected by smartphones. Missingness is an important aspect of analyzing aggregate trip survey behavior, yet geolocation missingness does not describe trip length error as a whole, as it has nothing to do with the error introduced by trip segmentation or location resampling algorithms.

Mode inference

Smartphone-based passive travel surveys either rely on a user prompted recall survey component or use travel diary

creation algorithms to assign modes to continuous mode-segmented trajectories, with trajectory segmentation error inherent to mode inference error. Prompted recall surveys can serve as an accurate method to collect labels for pre-segmented trips, but (i) they do not capture the error caused by the segmentation algorithms, and (ii) they rely on user labeling rates, which are highly variable (*30*). While attempting to automatically predict modes and reduce user burden, researchers have used decision trees (*31, 32*), hidden Markov models (*33, 34*), and neural networks (*35, 36*) to distinguish between travel modes. However, the algorithms either process unimodal trips (*31, 33, 34, 36*) or evaluate mode inference and section segmentation separately (*32, 35*).

Motivated by the work of (*22*), (*17*) introduce a methodology to evaluate continuous transportation mode segmentation by aligning temporal intervals. The temporal alignment procedure introduced in our paper builds off (*17*) by collapsing the complexity of spatial and temporal errors—precision and recall of matched segments, shift-in and shift-out penalties—into a single set of precision and recall metrics suitable for error propagation and downstream analysis. Our work is able to isolate differences in continuous mode segmentation by operating system and pipeline stage, which we show to play a significant role in our error metrics. Additionally, (*17*) test their data against a dataset of 26 users who installed the MEILI app and created user-annotated travel diaries with no ground truth for trajectories, whereas we are able to generate results against a high-quality dataset with predefined ground truth trip trajectories and transition times. Furthermore, both the dataset and source code used in our case study are open source, allowing for reproducibility and lowering the barrier for others to build off our work.

Preliminaries

This section provides the necessary preliminaries, formalism, and terminology used in the remainder of this paper. We fit the definitions and notations proposed in (*17, 37*) to be consistent with the formalisms of (*20*). In particular, we add formalisms and definitions for *base modes*, *segmentation decomposition*, *histories*, and *length measures*, while renaming *tripleg* as *section*. A comprehensive list of the notations introduced and used throughout this paper can be found in Table 1.

Modes, segment trajectories, and sections

Let $M = \{m_1, m_2, \ldots, m_k\}$ be an exhaustive set of *transportation modes*, and let $\{m_t, m_s\}$ be the transition and stop *mobility states*. Let BM be an exhaustive set of *base modes* and let the map $b : M \to BM$ be the *base mode map*. A mode m is said to *agree* with base mode bm if $b(m) = bm$.

Let $L = \langle l_1, l_2, \ldots l_n \rangle$ be a sequence of *segment trajectories* where $l_i = (x, y, t, m)$ denotes the i-th location in L. Let $l_i.x,\ l_i.y$ be the x and y location coordinates, $l_i.t$ denote the time at which $l_i.t$ is recorded, and $l_i.m \in M \cup \{m_t, m_s\}$ be the transportation mode.

Let a trip *section* be a sequence of points in temporal order where the point timestamps are between the section's start and end, the point's mode is valid, and all points have the same mode. Formally, a section is defined as the maximal length traveled in a single mode and overlaps a sequence of locations within the time period $[s; e]$, where s and e represent the start and end timestamp of a section, as shown in Equation 1. *Section length* is defined to be the sum distance between consecutive locations as in Equation 2, where $||\cdot||$ is the euclidean norm.

$$S^m = \langle l_i^{x,y,t,m} \mid (l_{i+1}.t > l_i.t) \wedge (l_i.t \in [s; e]) \wedge (l_i.m \in M) \wedge (l_i.m = l_{i+1}.m) \rangle \quad (1)$$

$$S^m.\ell = \sum_{i}^{n-1} ||l_i - l_{i+1}|| \quad (2)$$

Trips

Let a *trip* be a sequence of sections in temporal order, each with an associated mode, where the start of the trip is the start of the first section, the end of the trip is the end of the last section, and the sections are separated by zero-length modal transitions. Formally, let a trip be defined as the maximal length sequence of sections en route to a destination, where between every section S_i, S_{i+1} exists a period W_i^{i+1} in a transition state, m_t, as shown in Equation 3. W_i^{i+1} represents the transition period between sections of distinct modes and can be of length 0. It is important to note that the maximum transition period is strictly less than a segmentation algorithm's given *dwell time buffer*. *Trip length* is defined to be the sum section length, as in Equation 4.

$$T = \langle S_i^{m_i},\ W_i^{i+1} \mid (T.s = S_0.s) \wedge (T.e = S_n.e) \wedge (W_i^{i+1}.s = S_i.e) \wedge (W_i^{i+1}.e = S_{i+1}.s) \wedge (S_i.e \leq S_{i+s}.s) \rangle \quad (3)$$

$$T.\ell = \sum_{i}^{|T|} S_i^m.\ell \quad (4)$$

Mode-segmented trajectories and histories

A *mode-segmented trajectory* is a sequence of trips in temporal order, with strictly nonzero dwell times between trips, where the start of the trajectory is the start of the first trip, and the end of the trajectory is the end of the last trip. Formally, let a mode-segmented trajectory be defined as a sequence of trips, where between every trip T_i, T_{i+1} exists a period WT_i^{i+1} in a stop state, m_s, as shown in Equation 5. WT_i^{i+1} represents the *dwell time buffer* between two consecutive trips and is of length 0. The segment-decomposed trajectory is a trajectory where the

Table 1. Notations by order of introduction

Symbol	Name	Definition
M	Transportation modes	Set of transportation modes
BM	Base modes	Set of transportation base modes
L	Segment trajectories	Sequence of segment trajectories
S^m	Section	Sequence of consecutive segment trajectories with mode m
$S^m.\ell$	Section length	Sum distance between segment trajectories in S^m
W_i^{i+1}	Section transition period	Time interval between a sections S_i and S_{i+1}
T	Trip	Sequence of sections and transition periods
$T.\ell$	Trip length	Sum of section length of sections in T
WT_i^{i+1}	Dwell time buffer	Time interval between a trip T_i and T_{i+1}
G	Mode-segmented trajectory	Sequence of trips and dwell time buffers
N	Segment-decomposed trajectory	A mode-segmented trajectory with trips decomposed into section and section transition period form
$N.\ell$	Mode-segmented trajectory length	Sum of trip lengths in G
H	History	Sequence of segment-decomposed trajectories and dwell time buffers
$H.\ell$	History length	Sum of mode-segmented trajectory lengths of segment-decomposed trajectories in H
H^{gt}	Ground truth history	A history constructed from ground truth data
H^{inf}	Inferred history	A history constructed from inferred data
E_{S^m}	Energy impact	The energy impact corresponding to a section S^m
M^{GT}	Ground truth modes	Set of modes in ground truth data
M^{inf}	Inferred modes	Set of modes in inferred data

trips are replaced by their ordered constituent sections, as in Equation 6. *Mode-segmented trajectory length* is defined to be the sum of trip lengths, as in Equation 7.

$$G = \langle T_i,\ WT_i^{i+1} \mid (T_i.e < T_{i+1}.s) \\ \wedge (WT_i^{i+1}.s = T_i.e) \wedge (WT_i^{i+1}.e = T_{i+1}.s) \rangle \quad (5)$$

$$N = \langle S_{i,j}^{m_i}\ W_{i,j}^{i+1,j},\ WT_j^{j+1} \mid (S_{i,j}^{m_i} := S_j^{m_i}) \\ \wedge (W_{i,j}^{i+1,j} \in T_i) \wedge (T_i \in L) \wedge (WT_j^{j+1}) \in L \rangle \quad (6)$$

$$N.\ell = \sum_{i,j}^{|L|} S_{i,j}.\ell \quad (7)$$

A *history* is an ordered concatenation of trajectories, separated by dwell times, and decomposed into constituent sections and wait transitions. Formally, let a history be defined as the time-ordered union between the segment decomposition of a mode-segmented trajectory G and the ordered sequence of each WT_i^{i+1}, as shown in Equation 8. *History length*, $H.\ell$, is the sum of segment decomposed trajectory lengths, as in Equation 9.

$$H = \langle N_i,\ WT_i^{i+1} \mid (N_i.s < N_{i+1}.e) \\ \wedge (WT_i^{i+1}.s = N_i.e) \wedge (WT_i^{i+1}.e = N_{i+1}.s) \rangle \quad (8)$$

$$H.\ell = \sum_{i}^{|L|} N_i.\ell \quad (9)$$

Data and methodology

This section presents a methodology to measure trip length and mode inference, revolving around the comparison of ground truth histories H^{GT} and inferred histories H^{inf}. The methodology requires the use of a dataset with high-quality ground truth, including spatial trajectory ground truth and mode transition events.

MobilityNet: A high-quality ground-truthed dataset

The methodology assumes the existence of a high-quality, carefully collected dataset to provide ground truth. This paper uses MobilityNet (*38*), a public dataset containing information from three *artificial timelines* that cover 15 different travel modes (*39*). Each artificial timeline is a predefined sequence of trips with specified travel trajectories and modes. Data collectors complete these trips by strictly following timeline specifications while carrying multiple *evaluation phones* of varying operating systems and *sensor configuration settings*. The predefined trajectories provide spatial ground truth, and temporal ground truth is coarsely recorded by hand as data collectors transition between modes (e.g., get off the bus and begin to walk). Artificial timelines are critical since they (i) eliminate privacy concerns, (ii) provide an independent source of ground truth, and (iii) support multiple repetitions, all of which are baseline requirements for high-quality data collection. In the following sections, we denote both spatial and temporal ground truth simply as *ground truth*.

The MobilityNet dataset contains 1080 h of multimodal travel from three artificial timelines (*39*). Data collectors made *evaluation runs* for each artificial timeline using multiple *evaluation phones* with various configurations of virtual sensor settings. The setting configurations are selected from high- and medium-accuracy (`HA/MA`) (quality of data) settings, high- and medium-frequency (`HF/MF`) (quantity of data) settings, and duty cycling versus always-on (`DC/AO`) data collection settings. *Configurations* are a combination of phone settings, so `MAMFAO` stands for medium-accuracy, medium-frequency, always-on data collection.

These settings were used to configure the black-box fused location implementations on Android and iOS, which automatically combine GPS, Wi-Fi, cell towers, and Bluetooth signals to provide location points. These implementations are proprietary solutions that do not indicate when they switch between location sources. The dataset only contains the empirically observed values for each configuration. However, a reasonable assumption is that high-accuracy settings tend to favor GPS for collecting location data, whereas high-frequency settings will sense and process data more often. Duty cycling allows for high-accuracy and high-frequency settings without excessive battery drain, but has sensing gaps at trip start and end due to sensing delay.

Trip length metrics

Trip segmentation proposes a unique challenge to the direct analysis of trip length error. In particular, errors in trip segmentation can lead to double-counting trips. For instance, a pipeline may segment a ground truth trip consisting of a walk to a train station and a subway ride to the ballgame as separate trips. If we evaluate both walking and subway trips against the same ground truth trip to the ballgame, then we compute two errors for one trip! To avoid this scenario, we combine the inferred walking and subway trips into a single *minimal history*, H_m, so that there exists a direct correspondence between ground truth trips and error outputs. In general, we create minimal histories to ensure that one trip, either inferred or ground truth, is associated with only one relative error. We can then perform our analysis on minimal history length, with minimal history length serving as the linear combination of trip lengths (Eq. 7). Thus, trends in minimal history length error reflect trends in trip length error.

We propose a method of measuring minimal history length through standard metrics of measurement error: signed error and signed relative error. Algorithm 1 computes such signed error (lines 1–5) and signed relative error (line 6). It is important to note that these metrics hold little value when viewed on an individual basis. For instance, if we fail to infer a history, the resulting signed relative error of -1 does not provide any information about histories that were inferred.

However, analysis of an error *distribution* does provide valuable information. Mean signed relative error for a set of histories can expose trends in history length bias, and signed relative error variance shows how consistent these trends are. Studying outliers in a collection of histories can explain how often we observed uncharacteristic behavior.

Algorithm 1 measurement_error(H_m^{inf}, H_m^{GT})

Require: $H_m^{GT} \neq \emptyset$
1: **if** $H_m^{inf} = \emptyset$ **then** ▷ no inferred history
2: SE $\leftarrow -H_m^{GT}.\ell$ ▷ signed error is the ground truth length
3: **else**
4: SE $\leftarrow H_m^{inf}.\ell - H^{GT}.\ell$ ▷ signed error
5: **end if**
6: SRE $\leftarrow$ SE $/H_m^{GT}.\ell$ ▷ signed error relative to ground truth length
7: **return** SE, SRE

Mode inference metrics

On face value, mode inference neatly fits the common paradigms of multinomial classification. However, the continuity of mode-segmented trajectories provides a barrier to institute a classification regime in a unified way, as described in (*17*) (Sect. 4.1).

We propose a method of temporal interval alignment to remove this barrier and measure mode inference. The proposed method differs from the interval alignment method in (*17*) in that we temporally align *solely* based on interval overlap as defined in (*22*). Investigating temporal overlap allows for evaluation of *how often* a mode was correctly inferred, in accordance with the motivations of traditional multinomial classification problems (*40*).

The temporal align process takes an *ordered* inferred history and an *ordered* ground truth history and creates an *aligned* inferred history and an *aligned* ground truth history (Fig. 1).

Consider Fig. 1 with a ground truth history of walking and then riding a bicycle, with data collection that typically lags ground truth. Lagging causes the data collection to begin sensing after a delay, is late catching the transition between walking and biking, and registers another false walking section as the trip winds down, persisting for some time after the trip ends.

We want to determine when our mode segmentation algorithm correctly identifies when we were walking or biking while also capturing faulty inference (i.e., inferring walking when really biking). To do this, we align the ground truth and inferred history based on timestamps, padding the sections as necessary. Thus, we see where the inferred and ground truth histories overlap. In Fig. 1, time ranges (2) and (4) represent modal overlap, whereas (1), (3), (5), and (6) are the time ranges where there is no overlap and an error has occurred.

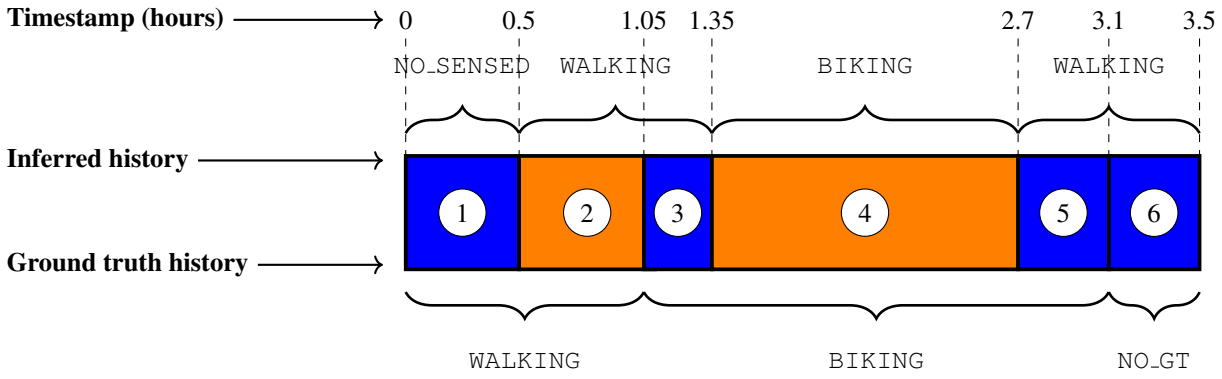

Figure 1. Example temporal history alignment for arbitrary inferred and ground truth histories in hours: (1) we begin walking but data collection has not started; (2) we are walking, and pipeline correctly infers walking; (3) we begin biking, but pipeline still infers walking; (4) we are biking, which the pipeline correctly infers; (5) we near the end of our biking trip, which pipeline infers as walking; (6) our trip has ended, but walking is still inferred for a short period of time.

At a high level, we construct *aligned* ground truth and inferred history so that the i-th element in the inferred history has the same time interval as the i-th element in the ground truth history. If both histories have the same number of elements, we trim each element to the longest common subsequence of aligned timestamps (e.g., elements 2, 3, 4 in Fig. 1). If the *ground truth* history has more elements (i.e., missing trips), we add "ghost" intervals, each with the mode `NO_SENSED` to the inferred history to align with the ground truth timestamps. If the *inferred history* has more elements (i.e., spurious trips), we add ghost intervals, each with the mode `NO_GT` to the ground truth history. We append `_START`, `_MIDDLE`, or `_END` to our ghost intervals in order to signify when during a trip a ghost interval is inserted. At the end of the alignment process, we have histories that are an ordered sequence of temporally aligned unimodal *elements*.

The pseudocode in Algorithm 2 details the temporal alignment process with respect to the temporal_align(·)-function, which takes in two arguments: an *ordered* inferred history H^{inf} and an *ordered* ground truth history H^{GT}. Each element of the histories is either a segment or a dwell time buffer, each equipped with a mode and interval timestamps. The function can be thought of as two successive subprocesses: the first padding the start and end of each history so they exist in the same time interval (lines 1–15), and the second creating two aligned histories, H_a^{inf} and H_a^{GT}, so that the i-th entry in H_a^{inf}, $h_{i_a}^{inf}$ has the same interval as the i-th entry in H_a^{GT}, $h_{i_a}^{GT}$, while each $h_{i_a}^{inf}$ and $h_{i_a}^{GT}$ remains unimodal and reflects the padded histories' temporal mode inferences (lines 16–32).

Subprocess 1, history padding: Check if one of the inferred or ground truth histories is empty, and if so, replace with a history of length and duration 0 (lines 1–6). Pad the start and end of each history with dwell time buffers so that both histories have the same start and end timestamps (lines 7–18). This subprocess has complexity $\Theta(1)$.

Subprocess 2, history alignment: Cycle through each element of inferred history h_i^{inf} (line 20), searching for overlap with an element of ground truth history h_j^{GT} (line 24). Once found, construct an aligned history element h_a with start and end timestamps consistent with the overlap interval. Then, append h_a to the aligned inferred history with mode $h_i^{inf}.m$ and to the ground truth history with mode $h_j^{GT}.m$ (lines 25–28). The construction of aligned histories ensures they remain unimodal ordered sequences so that any timestamp in the aligned history is either in the corresponding history or not in the corresponding history with a ghost mode. This subprocess had complexity $\mathcal{O}(n+m)$ where $n = |H^{inf}|$ and $m = |H^{GT}|$ as $|H_a^{inf}| \leq |H^{inf}| + |H^{GT}|$.

Results

In this section, we illustrate how our evaluation methodology can be used to characterize the error of the raw data, and how it changes as it moves through multiple stages of a multistep postprocessing pipeline. We also demonstrate how the approach can be used to compare multiple algorithms for the same task. We use the National Renewable Energy Laboratory's (NREL's) open-source OpenPATH (formerly emission) pipeline as an example of the types of algorithms used for travel diary creation. We sketch the OpenPATH algorithms as necessary to illustrate the challenges in defining metrics, but we do not elaborate on them. The methods defined here can be used to evaluate any set of travel diary creation algorithms, even a black-box one that randomly generates segments and modes.

Algorithm 2 temporal_align(H^{inf}, H^{GT})

Require: *ordered* H^{inf}, H^{GT} and at least one of H^{inf}, H^{GT} be non-empty

if $H^{inf} = \emptyset$ **then** ▷ create ghost inferred history
  $H^{inf} \leftarrow \langle T_1, WT_1^2, T_2 : T_1.s = H^{GT}.s \wedge T_1.m = m_s \wedge T_2.e = H^{GT}.e \wedge T_2.m = m_s \rangle$
end if
if $H^{GT} = \emptyset$ **then** ▷ create ghost ground truth history
  $H^{GT} \leftarrow \langle T_1, WT_1^2, T_2 : T_1.s = H^{inf}.s \wedge T_1.m = m_s \wedge T_2.e = H^{inf}.e \wedge T_2.m = m_s \rangle$
end if
$s' \leftarrow H^{inf}.s - H^{GT}.s$ ▷ start misalignment
$e' \leftarrow H^{inf}.e - H^{GT}.e$ ▷ end misalignment
if $s' > 0$ **then** ▷ inferred start is later - i.e., t = 2 vs. t = 1
  $H^{inf} \leftarrow \langle T_0, WT_0^1, T_1, \ldots T_n : T_0.s = |s'| \wedge T_1, \ldots, T_n \in H^{inf} \rangle$ ▷ pad start of inferred
else if $s' < 0$ **then**
  $H^{GT} \leftarrow \langle T_0, WT_0^1, T_1, \ldots T_n : T_0.s = |s'| \wedge T_1, \ldots, T_n \in H^{GT} \rangle$ ▷ pad start of GT
end if
if $e' > 0$ **then**
  $H^{GT} \leftarrow \langle T_1, \ldots T_n, WT_n^{n+1}, T_{n+1} : T_{n+1}.e = |e'| \wedge T_1, \ldots, T_n \in H_a^{GT} \rangle$ ▷ pad end
else if $e' < 0$ **then**
  $H^{inf} \leftarrow \langle T_1, \ldots T_n, WT_n^{n+1}, T_{n+1} : T_{n+1}.e = |e'| \wedge T_1, \ldots, T_n \in H_a^{inf} \rangle$ ▷ pad end
end if
$j \leftarrow 1$
for $i \leftarrow 1, \ldots, |H^{inf}|$ **do** ▷ loop over inf hist.
  while $j \leq |H^{GT}|$ **do** ▷ search every overlapping GT element
    if $h_j^{GT}.e < h_i^{inf}.s$ **then** ▷ GT element before inferred element
      $j \leftarrow j + 1$ ▷ keep searching
    else if $h_i^{inf}.e \geq h_j^{GT}.s$ **and** $h_i^{inf}.s \leq h_j^{GT}.e$ **then** ▷ inf & GT overlap
      $h_a \leftarrow \mathbf{0}$
      $h_a.s \leftarrow \max(h_i^{inf}.s, h_j^{GT}.s)$ and $h_a.e \leftarrow \min(h_i^{inf}.e, h_j^{GT}.e)$ ▷ overlap interval
      H_a^{inf}.append$(h_a : h_a.m = h_i^{inf}.m)$ ▷ append to inf with inf mode
      H_a^{GT}.append$(h_a : h_a.m = h_j^{GT}.m)$ ▷ append to inf with inf mode
      $j \leftarrow j + 1$ ▷ next GT entry
    else ▷ Check if the next inf and prev. GT entry have overlap
      if $i + 1 \leq |H^{inf}|$ and $j \geq 2$ and $h_{i+1}^{inf}.s < h_{j-1}^{gt}.e$ **then**
        $j \leftarrow j - 1$ ▷ overlaps, so move to prev. GT entry
      end if
      break ▷ Check next inf entry
    end if
  end while
end for
return H_a^{inf}, H_a^{GT}

NREL OpenPATH pipeline overview

OpenPATH is an open-source, extensible system confronting the challenge of transportation decarbonization in an interdisciplinary manner. OpenPATH instruments travel behavior, using background sensor data from a smartphone app to create individual travel diaries, which are then annotated by the user. It uses these travel diaries to calculate an individual carbon footprint, which are aggregated to generate energy or emission impacts. The energy calculations are shown in Equation 10, where H is a history of trips T with length $T.\ell$, trips consist of unimodal sections S^m with length $S^m.\ell$, and each mode m has a corresponding energy impact E:

$$\sum_{T \in H} \sum_{S^m \in T} E_{S^m} \times S^m.\ell \tag{10}$$

OpenPATH utilizes a **sensor-based multistep pipeline** (*41*) for processing phone sensor data and generating trip and sections labeled with automatically inferred modes. The pipeline is a sequence of independent deterministic stages (Fig. 2) so that the input to one stage is the output of another. The input stage is the data received from the

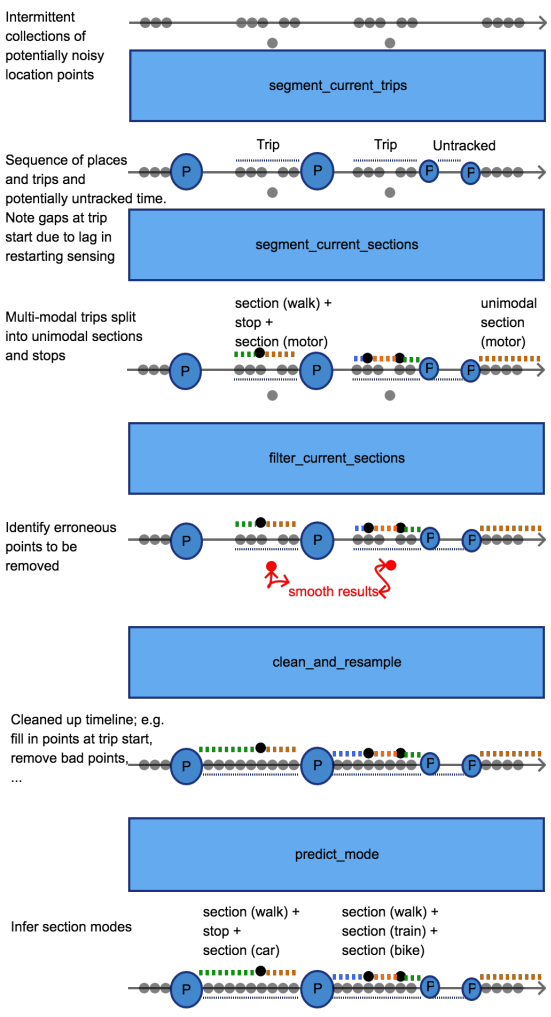

Figure 2. Sensor-based multistep OpenPATH pipeline as seen in (*20*) Fig. 5.1.

OpenPATH smartphone app, which takes advantage of a particular phone's built-in black-box virtual sensors.

These input data should serve as the baseline for any evaluation, but since they are received from phone sensors, they are point-based. To ensure that the data fit into our evaluation methodology, we also define a separate naive, lightweight, change-point detection-based algorithm that reflects the error characteristics of the underlying sensor data to the extent possible.

An analysis of the power vs. accuracy trade-off for these settings (*20*) (Ch. 7.2) shows that power consumption is primarily affected by data collection frequency on Android and by data collection accuracy on iOS. As a result, OpenPATH currently selects `HAMFDC` as the configuration for Android phones and `HAHFDC` for iOS phones. We refer to these sensor configurations as the *selected configuration settings*.

We evaluate the OpenPATH pipeline at different stages and on phones with varying sensor configurations. Our results show that the evaluation method is able to:

Distinguish performance of similar algorithms:
Inference based on an integration with map features results in weighted F_1-scores of 0.60 (iOS) and 0.74 (Android), outperforming random-forest-based mode inference with weighted F_1-scores of 0.25 (iOS) and 0.29 (Android).

Identify the change in error across pipeline stages:
Naive interpretation of geospatial phone data results in variance of distance errors of 0.217 (Android) and 0.0848 (iOS), whereas postprocessing decreases these errors to .0110524 (Android) and 0.0407 (iOS). Similarly, we see that naive mode inference results in weighted F_1-scores of 0.738 (Android) and 0.779 (iOS), which increase to 0.749 (Android) and 0.906 (iOS) after postprocessing.

Trip length

We use the measurement_error(·)-function in Algorithm 1 to calculate the distribution of signed relative error for a set of minimal histories.

Empirical evaluations of trip length suggest that the OpenPATH pipeline has an effective anomaly detection method at the `clean_and_resample` pipeline stage, while also having a systematic bias toward underestimation with little influence from random error.

Performance analysis We evaluate trip length at the sensor-based input to the pipeline, minimally processed to fit our evaluation methodology (naive) and at the output of the `clean_and_resample` (clean) OpenPATH pipeline stage.

Table 2 has the sample mean, variance, min, max, and size of the signed relative error distribution at selected sensor settings. We also notice that both phone operating systems have a small variance and negative sample mean at the clean output, indicating a systematic underestimation of trip length.

Figure 3 shows the box plot for the signed relative trip length error on phones running selected configuration settings. We first notice that the signed relative error distribution on both operating systems has a significantly narrower range at the clean and resample stage as opposed to raw output. This implies that the trajectory smoothing algorithm is effective at correcting trip overestimation anomalies. However, we see that the clean and resample stage has no capability to address underestimating trip length errors.

Discussion There are two main sources leading to underestimation of trip length in the raw MobilityNet data:

- The sensed points are underreported (Fig. 4). These errors cause poor resolution of trip trajectory, leading

			$\bar{x}$	s^2	min	max	n
naive	Android	HAMFDC	0.197	0.217	-0.321	2.183	32
	ios	HAHFDC	-0.0608	0.0848	-1.0	0.419	42
clean	Android	HAMFDC	-0.0777	0.01105	-0.4434	0.0696	32
	ios	HAHFDC	-0.1397	0.0407	-1.0	-0.001677	42

Table 2. Statistics for signed relative error trip length distribution

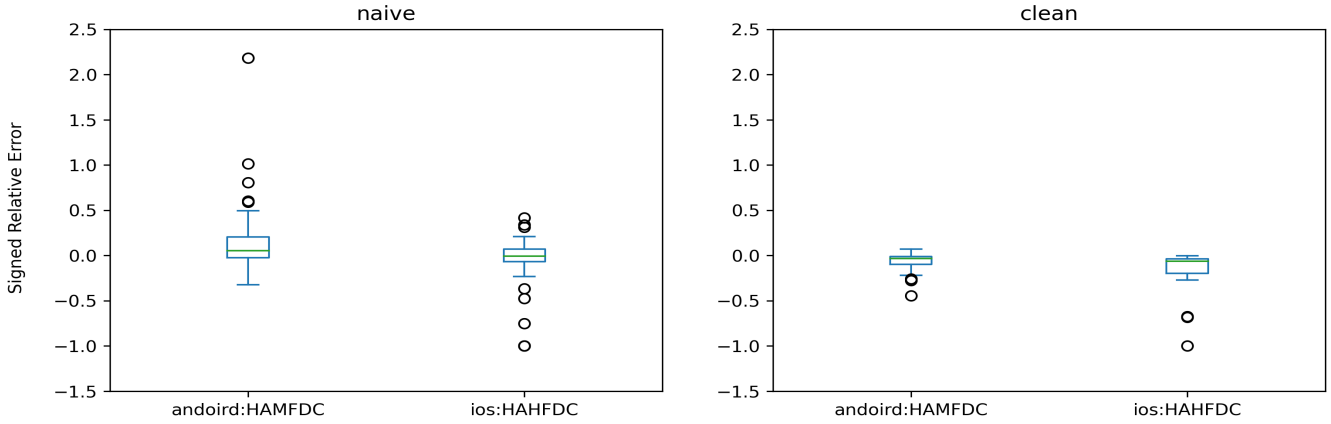

Figure 3. Signed relative error box plots

to an underestimation of trip length because more of the trip is inferred to be traveled in a straight line, in opposition to ground truth, which has more detail and thus has points offset from the inferred trajectory. Notice that this error is the converse of the error caused by spatial offset of sensed points.

- The sensed points are collected at a delay (Fig. 5). These errors tend to cause a shorter time range for data collection, thus leading to a shorter trip length, all else being equal.

This contrasts with the two main sources of overestimation:

- The sensed points are spatially offset from the real trajectory (Fig. 6, top). The green points in the figure represent the ground truth location points; the ground truth trajectory is the piecewise join of these points. As we can see, when the road curves, the points are very close together so that the piecewise joins represent a curve. If the sensed points are spatially offset, as in the purple, brown, and red points in the figure, the resulting errors will always represent an overestimation. This is because the ground truth distance between two points, which we compute as a straight line, will always be shorter than the inferred path between two points that detours to include the spatially offset point.
- The sensed points have temporal inconsistencies (Fig. 6, bottom). These errors will lead to overestimation, as inferred trip length will repeatedly zigzag between the current point and an earlier one, with each repetition adding an additional distance to total trip length.

Furthermore, it is clear to see how both overestimation and underestimation errors can occur in the same trip—e.g., we may start a trip at a delay (underestimation) and then have spatial offset for our data collection points (overestimation). However, we observe that the MobilityNet dataset has a systematic bias toward underrepresenting trip length (Table 2), implying that our overestimation errors hold less impact in our total trip length calculations. Note that it might be possible to compensate for these errors through an improved postprocessing pipeline. However, with this paper we aim to demonstrate how we can rigorously quantify the impact of any postprocessing improvement.

Mode inference

We evaluate the mode inference error as a multinomial classification problem, building confusion matrices to visualize classifier predictions and calculating F_1-scores to measure classifier performance, both for individual base modes and in aggregate.

Evaluations of mode inference show that rule+Geographic Information Systems (GIS)-based mode inference drastically outperforms random forest mode inference at the `predict_mode` pipeline stage. Part of the random forest's underperformance can be explained by the training data, which have poor allocation of support and rely on recruiting a small number of collectors where ground truth is reported by prompted recall (Table 1 in (*42*)).

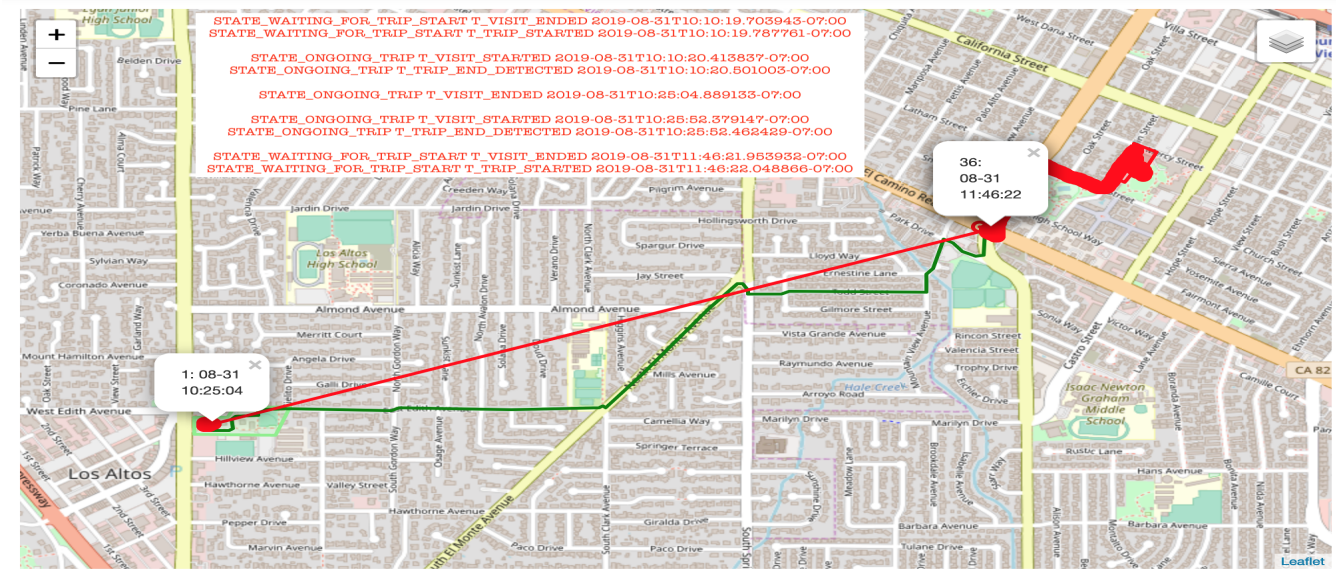

Figure 4. Example of bad data causing lingering missing sections. There are two ground truth trips, one from `10:07:27 -> 10:23:08` and another from `11:30:50 -> 11:52:38`. The sensed data include an ephemeral trip from `10:10:19 -> 10:10:20` with no points and from `10:25:04 -> 10:25:52` with some points. Then, most of the return trip is missed before getting a trip start at `11:46:21`. These trips are merged into one unimodal section in the postprocessing `10:25:07 -> 11:54:17`.

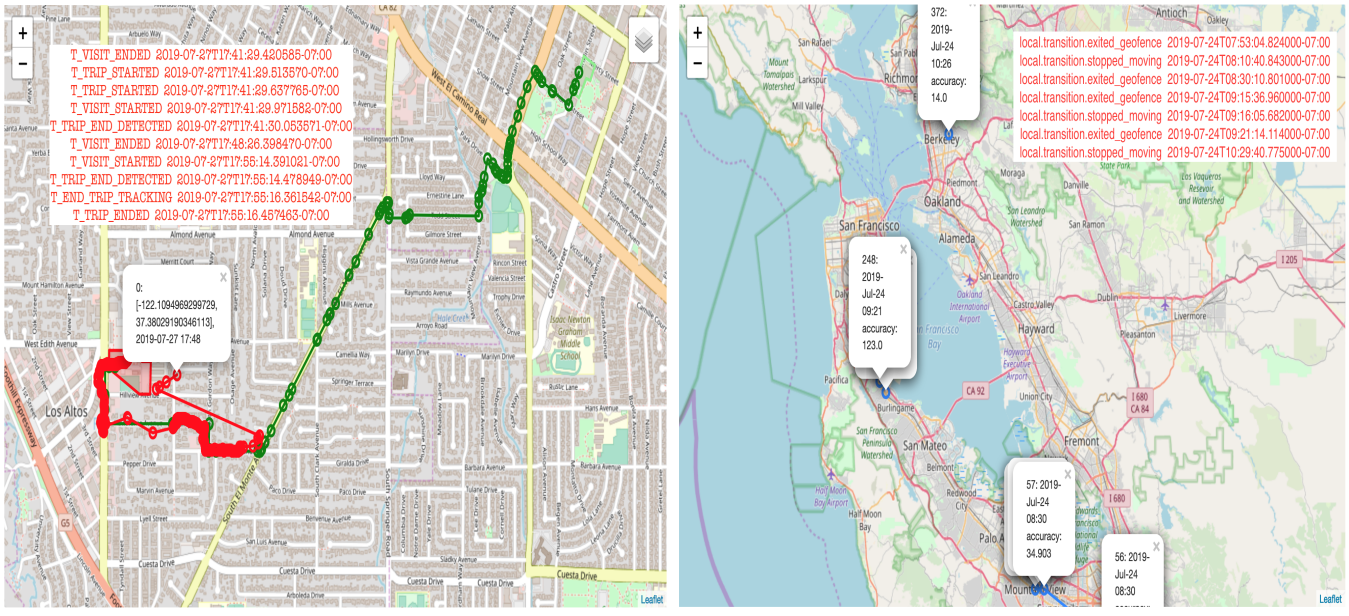

Figure 5. Examples of errors in segmentation captured by the segmentation metrics.
Left – large error in the trip start time for an iOS `HAHFDC` run: The green line is the ground truth, and the red line is the sensed data. We received a visit end (trip start) transition at 17:41, but we detected a trip end within 30 ms, so we did not sense any data. The next trip start was at 17:48, when we did start reading values, but this was almost the end of the trip.
Right – error in segmentation trip counts for an Android `HAMFDC` run: On one multimodal trip from Mountain View to Berkeley, we get multiple trip start and end transitions, largely corresponding to transit transfers. Note that at 08:30, there are two consecutive geofence exits (08:30 and 09:15) and an erroneous point off the map.

Confusion matrix The construction of confusion matrices for discrete classification problems is well defined and implemented (*43*). We adapt this methodology to our continuous problem, recording duration of mode overlap in

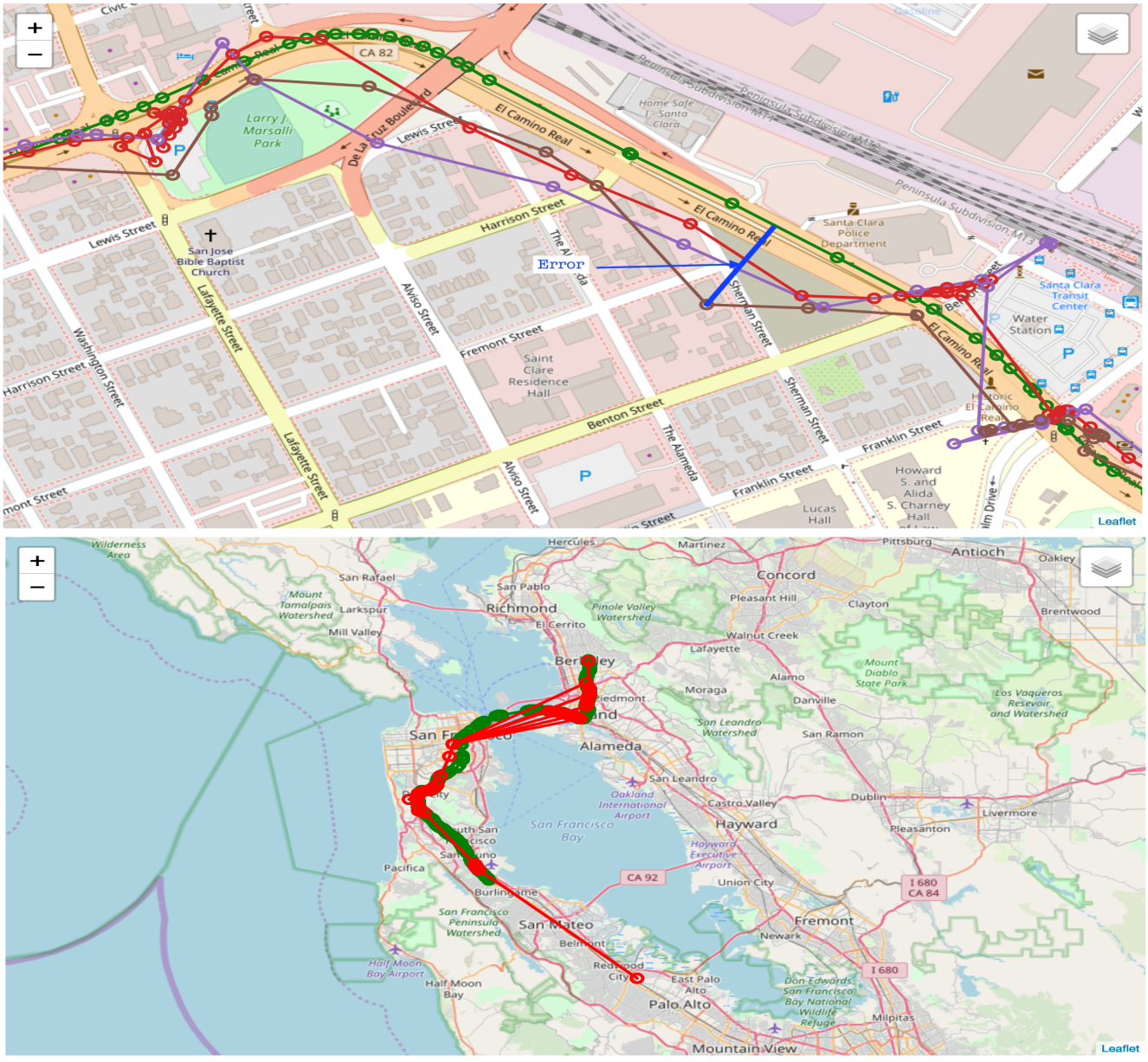

Figure 6. Examples of errors in trajectory.
Top – Spatial errors from iOS `MAHFDC` runs: The green line is ground truth, and other colors are the sensed data from three separate data collection runs along the same trajectory. The spatial error is the shortest perpendicular distance to the spatial ground truth line (i.e., the thick blue line from the brown point to the green line).
Bottom – Temporal errors from an Android `HAMFDC` run: The sensed points in red are largely along the spatial ground truth trajectory in green, but they periodically return to a previous point in the trajectory, generating zigzags, each having their own distance. It is important to note that the black-box geolocation data collection implemented by Android and iOS inhibit the qualification of such error sources.

each matrix entry in lieu of the traditional classification "hit" count.

The pseudocode in Algorithm 3 details the construction of a confusion matrix for an inferred and associated ground truth history. Each column of the matrix represents an inferred mode (line 1), and each entry represents the total time spent in an aligned ground truth mode (lines 3–8), where M^{GT} is a set of ground truth modes and M^{inf} is a set of inferred modes.

Multinomial classification Temporal alignment allows for the construction of the *continuous* multinomial classification process, which is detailed in relation to the pseudocode in

Algorithm 3 confusion_matrix(H^{inf}, H^{GT})

Require: *ordered* H^{inf}, H^{GT} **and** $H^{GT} \neq \emptyset$
1: $CM_{|M^{GT}|,|M^{inf}|} = \mathbf{0}$
2: H_a^{inf}, H_a^{GT} = temporal_align(H^{inf}, H^{GT})
3: **for** $i \leftarrow 0, \ldots, |H_a|$ **do** ▷ either history, since they are aligned
4: duration = $h_{a_i}.e - h_{a_i}.s$
5: $i \leftarrow$ index of $h_{a_i}^{GT}.m$ in M^{GT} ▷ index of the ground truth mode
6: $j \leftarrow$ index of $h_{a_i}^{inf}.m$ in M^{inf} ▷ index of the inferred mode
7: $cm_{i,j} \leftarrow cm_{i,j} +$ duration ▷ add duration to confusion matrix entry
8: **end for**
9: **return** CM

Algorithm 4. The classification_metrics(·)-function takes in three arguments: an *ordered* inferred history, *ordered* ground truth history, and a base mode map. The base mode map allows for a standardized method of comparing histories of different modal sets. For instance, a set of ground truth modes might be {`WALKING, BIKING, E_SCOOTER`} while the set of inferred modes is {`WALKING, BIKING`}. In this scenario, the base mode map would take {`BIKING, E_SCOOTER`} to {`BIKING`}, the associated base mode. Classification is performed via a one-versus-rest qualification system, where an instance of a given class is positive and an instance of any other classes negative, where classifications are made for every class. For our problem, an *instance* is the i-th element of an aligned inferred or ground truth history. The classification_metrics(·)-function must first align the histories (line 1), after which it begins cycling through a set of base modes (lines 2 and 3), and for each base mode, cycles through instances of aligned histories (line 4). The duration of the element is calculated (line 5), and conditionally added to one of the following modal classification tallies (lines 6–14):

TP^{bm} if the inferred mode and ground truth mode both agree with the base mode.

FP^{bm} if the inferred mode agrees with the base mode but the ground truth does not.

FN^{bm} if the inferred does not agree with the base mode but the ground truth mode does.

TN^{bm} if neither the inferred mode nor the ground truth mode agree with the base mode.

Outputs of the classification_metrics(·)-function can be used to calculate the F_1-score of a base mode, as shown in Equation 11.

$$F_1^{bm} = \frac{2 \cdot TP^{bm}}{2 \cdot TP^{bm} + FN^{bm} + FP^{bm}} \tag{11}$$

The weighted F-score is computed by taking the average of F-scores for each base mode weighted by the support of that base mode, which is the sum of confusion matrix row sums for each mode that maps to a base mode, $M_{bm} = \{m : b(m) = bm, m \in M\}$, as in Equation 12.

$$F_1^{avg} = \sum_{i=1}^{|BM|} \left[\sum_{j=1}^{|M_{bm_i}|} \sum_{k=1}^{|M^{inf}|} cm_{j,k} \right] \cdot F_1^{bm} \tag{12}$$

Sketches of analysis algorithms We evaluate the mode classification results at multiple stages of the OpenPATH pipeline, with two implementations for each stage.

- baseline phone inputs, minimally processed to fit out evaluation methodology (Naive)
- `analysis/cleaned_trip`, `analysis/cleaned_section` (Clean)
  - master: combine motion activity + location + transitions + Airplane/High Speed Rail (HSR) detection
  - GIS: master + flip-flop smoothing
- `analysis/cleaned_trip`, `analysis/inferred_section` (Inferred)
  - random forest
  - rule+GIS based

Naive: The naive algorithm is intended to segment raw, point-based sensor data from the phone so that it fits into our evaluation methodology and can be used as the baseline. At a high level, this algorithm finds start and end change points and detects ranges as matched pairs of start and end transitions. The algorithm detects *trip* change points via virtual sensor triggers (e.g., geofencing, `VISIT` start, `VISIT` end, and dwell time), and *section* change points when there is a change in phone-detected motion activity (e.g., `WALKING` to `BIKING`). So, the supported modes mirror the activity detection APIs on both Android and iOS and consist of active modes and a single motorized mode.

The naive algorithm ignores repeated transitions (e.g., `START, END, END`) correctly, and uses the ground truth of the last trip section to determine whether `STILL` activities represent a transition. We have

Algorithm 4 classification_metrics(H^{inf}, H^{GT}, b)

Require: *ordered* H^{inf}, H^{GT} **and** $H^{GT} \neq \emptyset$
H_a^{inf}, H_a^{GT} = temporal_align(H^{inf}, H^{GT})
$BM = \{b(m) : m \in M^{inf} \cup M^{GT}$ ▷ construct base mode set
for $bm \in BM\}$ **do** ▷ loop over base modes
  for $i \leftarrow 0, \ldots, |H_a|$ **do**
    duration $= h_{a_i}.e - h_{a_i}.s$
    if $bm = b(h_{a_i}^{inf}.m)$ and $bm = b(h_{a_i}^{GT}.m)$ **then** ▷ inf and GT in basemode
      $TP^{bm} \leftarrow TP^{bm} +$ duration ▷ true positive
    else if $bm = b(h_{a_i}^{inf}.m)$ and $bm \neq b(h_{a_i}^{GT}.m)$ **then** ▷ inf in basemode; GT is not
      $FP^{bm} \leftarrow FP^{bm} +$ duration ▷ False positive
    else if $bm \neq b(h_{a_i}^{inf}.m)$ and $bm = b(h_{a_i}^{GT}.m)$ **then** ▷ inf not in basemode; GT is
      $FN^{bm} \leftarrow FN^{bm} +$ duration ▷ False negative
    else
      $TN^{m} \leftarrow TN^{m} +$ duration ▷ True negative
    end if
  end for
end for
return $\left\{\left(TP^{bm},\ FP^{bm},\ FN^{bm},\ TN^{bm}\right) : bm \in \{b(m) : m \in M^{inf} \cup M^{GT}\}\right.$

intentionally eschewed any additional complexity to allow the error characteristics of the underlying sensor data to be reflected to the extent possible.

Clean: Clean modes are based on prior raw modes, which in turn are also based primarily on the activity detection API from the phone. They augment the motion activity data with location and transition data to improve robustness and also support Airplane (AIR) and high-speed rail (HSR) detection for long and/or fast trips. Of the two algorithms, master simply works with combined data, while GIS also attempts to smooth out "flip-flopping," in which the detected activity quickly changes between two modes. Clean *trips* also attempt to compensate for sensing delay at the start of a trip by extrapolating from the previous trip end.

Inferred: Random-forest-based mode inference is the traditional option, with a feature set largely based on (*32*) and detailed in (*42*) and (*44*). The model distinguishes between six modes: `WALKING`, `BIKING`, `BUS`, `CAR`, `TRAIN`, `AIR_OR_HSR`. It requires a large reference dataset for training, and once trained it can perform the computations locally and fast.

The rule+GIS-based mode inference algorithm requires access to an external service and is subject to all the reliability and robustness challenges that remote communication entails. It distinguishes nonmotorized modes based off speed. Motorized modes are inferred by querying OpenStreetMap (OSM) using the Overpass API (*45*) to determine whether end points coincide with transportation stops, with additional checks being used to reduce false positives. The use of OSM routes add support for `SUBWAY`, `TRAM`, `LIGHT_RAIL` while still distinguishing between `WALKING`, `BIKING`, `BUS`, `CAR`, `TRAIN`, `AIR_OR_HSR`.

Performance evaluation Figure 7 shows that rule+GIS-based mode inference outperforms random forest mode inference in every base mode. This is reflected in the weighted F_1-scores (Eq. 12), in which GIS-based mode inference scores 0.71 and 0.61 for Android and iOS selected configuration settings, whereas random-forest-based mode inference scores 0.28 and 0.21.

Figures 8 and 9 show the confusion matrices at various pipeline output stages on Android and iOS phones running the selected configuration settings. The confusion matrices for random-forest-based mode inference show that the random forest has a bias toward predicting a train trip as a car ride. Figure 11 in (*44*) can help explain this phenomenon, displaying the support of the random forest training set, with a noticeable overrepresentation of `CAR` for the `AUTOMOTIVE` base mode.

Not surprisingly, the bias toward `CAR` is not reflected in the confusion matrices for GIS-based mode inference. However, these confusion matrices show there is still some work to be done in distinguishing between automotive travel modes (`CAR`, `BUS`, `TRAIN`, `SUBWAY`). We hypothesize that this error may be relieved in part by snapping data collection points to road and track networks. From there, the pipeline can discern if a trip followed bus, train, subway, or car routes and better distinguish between the modes. We observe that we miss the walking portion of our trip in the more sophisticated algorithms (clean, random forest,

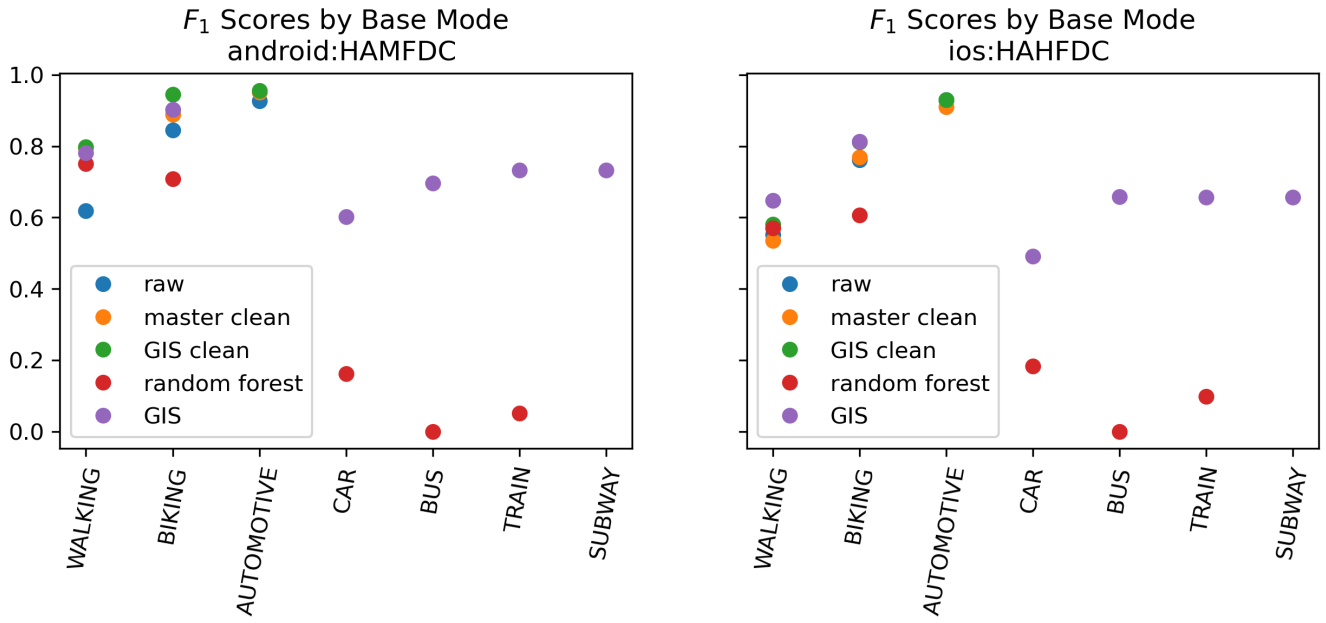

Figure 7. F_1-scores by base mode for selected operating system setting configurations

and GIS) 19% of the time on Android and 20% of the time on iOS phones with selected configuration settings. However, this percentage is 15% for Android and 16% for iOS at the baseline naive algorithm (Figs. 8, 9). This leads us to believe that the `clean_and_resample` pipeline stage is responsible and may identify and remove too many extraneous points.

Also notice that at every pipeline stage, phones running iOS predict `WALKING` the majority of the time when there is no ground truth in the middle of a trip, whereas Android phones do not. This suggests that iOS' black-box motion detection API has trouble discerning the transition between walking and stopping states.

Discussion We detail through example the main sources of mode inference error on the OpenPATH pipeline:

- The sensed points are collected at a delay (Fig. 5, left). The temporal alignment process allows for these errors to be captured at the start and end of a trip through history padding, and during a trip by constructing aligned history elements. In our confusion matrices, Figs. 8 and 9, the sensing delay at the start of a trip is classified under `NO_SENSED_START` while the error at the end of a trip is classified under `NO_GT_END`. During the middle of a trip, this error is captured as general overlap error and contributes to our FP and FN classification errors.
- The sensed points are incorrectly identified as a section transition (Table 3) or trip transition (Fig. 5, right). Again, this is captured via the temporal alignment process as FP and FN classification error.
- The sensed points are underreported (Fig. 4), which can lead to mode inference error, as the prediction algorithm has significantly less data to work with in determining mode, and thus may be more susceptible to faulty predictions. However, this error is not always captured by temporal alignment, as the prediction algorithm may guess correctly and the interval timestamps remain relatively accurate. However, this error will lead to underestimation of trip length, as discussed above, motivating the utility of using temporal alignment and trip measurement error in conjunction.

Conclusion and future work

This paper presents a novel approach to evaluate multistep sensor-based pipelines, where trip length is evaluated through standard methods of measurement error, and mode inference is evaluated using temporal alignment. Our method of temporal alignment is based on alignment proposed by (*17*), which is in turn based on time algebra alignment proposed by (*22*). The use of temporal alignment lets continuous mode inference fit classic discrete multinomial classification paradigms, allowing for a measurement of when in time a mode prediction is aligned with the ground truth. Additionally, our methodology allows for seamless cross-sectional and longitudinal studies on arbitrary pipelines.

The modularity of our evaluation methodology also supports expanding on current error models and distributions. Trip length measurements may take into account missingness factors as classified by (*29*). Temporal alignments might be

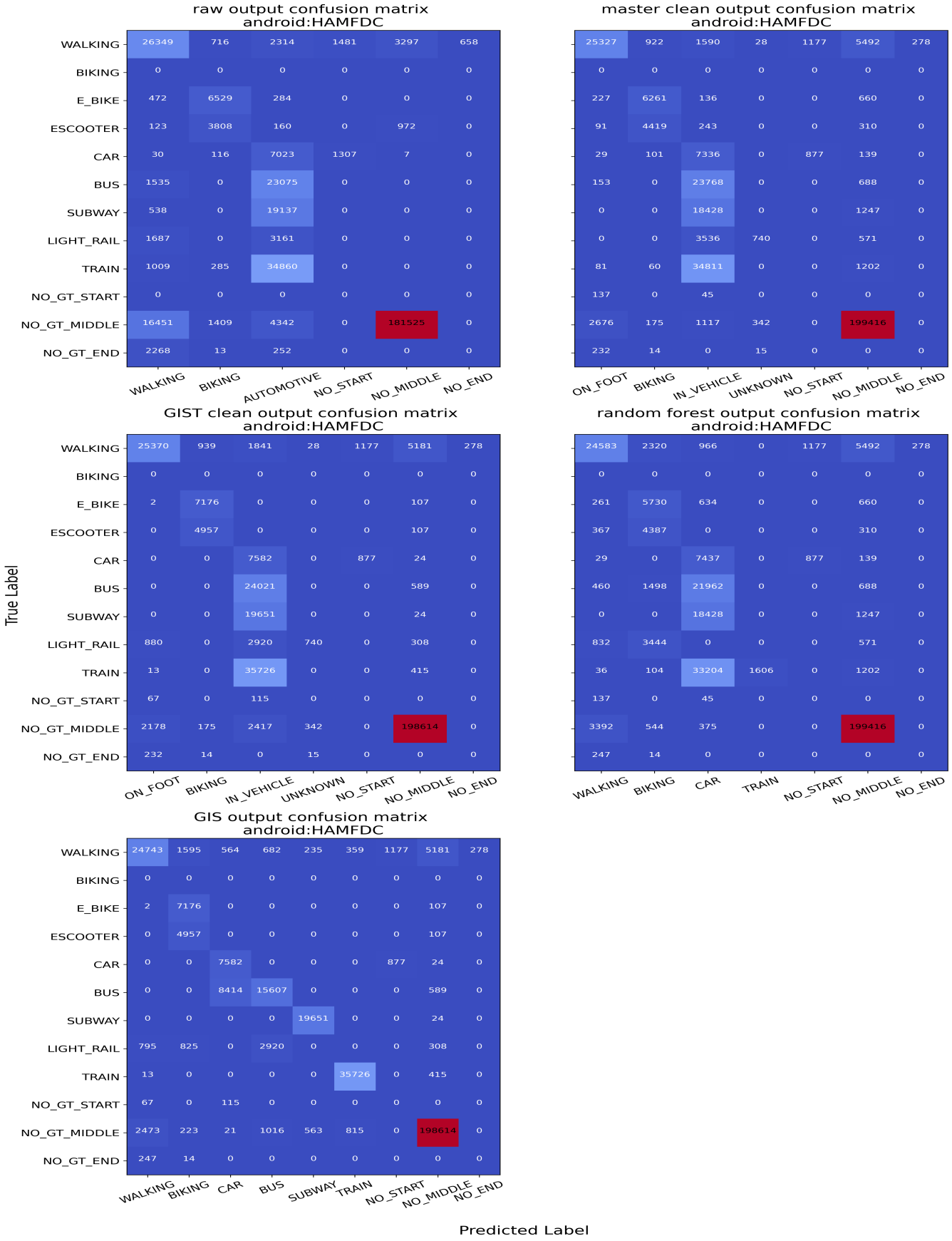

Figure 8. Confusion matrices at each pipeline output stage on phones running Android for selected setting configurations

Prepared using TRR.cls

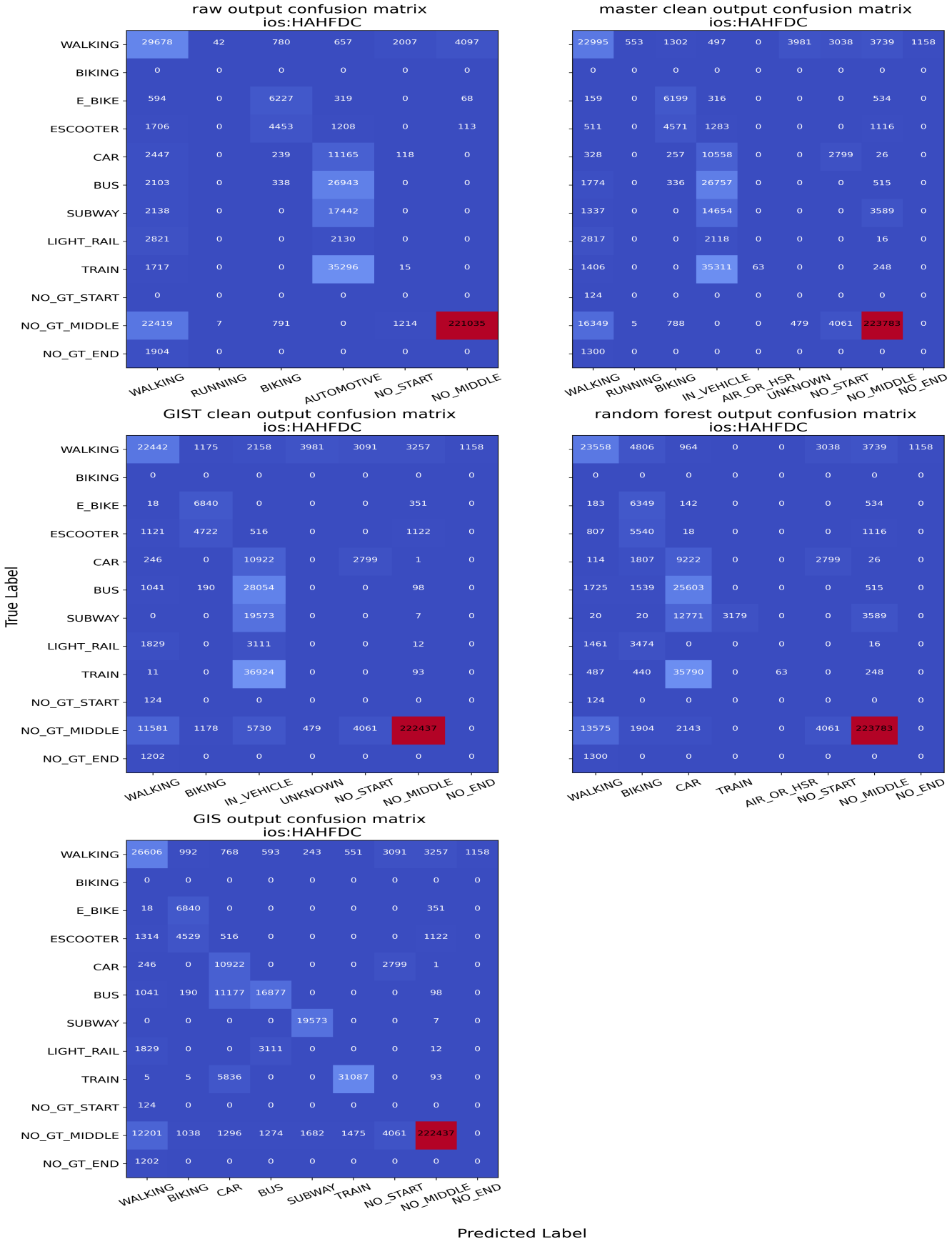

Figure 9. Confusion matrices at each pipeline output stage on phones running iOS for selected setting configurations

Prepared using TRR.cls

	automotive	confidence	cycling	running	stationary	walking	fmt_time
154	False	medium	False	False	False	**True**	19:01:53-07:00
155	False	high	False	False	False	False	19:02:46-07:00
156	False	medium	False	False	False	**True**	19:02:51-07:00
157	False	high	False	False	False	**True**	19:03:46-07:00
				...			
172	False	medium	False	False	False	**True**	19:17:59-07:00
173	False	high	False	False	False	**True**	19:18:15-07:00
174	False	high	False	False	False	False	19:19:06-07:00
175	False	medium	False	False	False	**True**	19:19:34-07:00
176	False	high	False	False	False	False	19:19:41-07:00
177	False	medium	False	False	False	**True**	19:19:49-07:00
178	False	high	False	False	False	**True**	19:20:04-07:00
179	False	high	False	False	False	False	19:21:16-07:00
180	False	high	False	False	False	**True**	19:21:36-07:00
181	False	high	False	False	False	False	19:22:36-07:00
182	False	high	False	False	True	False	19:27:21-07:00
183	False	high	False	False	False	False	19:28:01-07:00

Table 3. Example of how bad segmentation can lead to classification errors using an example from an iOS `MAHFDC` run. This trip consisted of a `walk_start` section from `18:59:17 -> 19:01:06`, a `suburb_bicycling` section from `19:01:06 -> 19:20:31` and a `walk_end` section from `19:20:31 -> 19:20:57`. However, the sensing API did not detect any cycling (see transitions above), so the only sensed section was `19:01:53 -> 19:27:21, WALKING`.

used to record distance, speed, or any arbitrary metric during overlap.

There are many avenues for future work to utilize or improve on our methodology and subsequent evaluation results. Most pressingly, oversegmentation penalties can be incorporated into our mode inference evaluation metrics to support count-based downstream metrics in addition to the current distance-based approach. Furthermore, the statistical significance of our evaluation can be improved by adding trips to the MobilityNet dataset, focusing on geographically diverse collection locations. We can use our methodology to study the factors that correlate with trip length error (mode, speed, trip duration, etc.). We can also construct an algorithm combining the random forest and rule+GIS-based algorithms and evaluate its performance against current methods of mode inference. Whatever the case for future work may be, the incorporation of our methodology will allow for sensor-based multistep pipelines output, in particular the interplay between trajectory segmentation and mode inference, to be understood in a continuous manner.

Acknowledgements

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. This work was supported in part by the DOE Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internship (SULI) program. This material is also based upon work supported by the DOE Office of Energy Efficiency and Renewable Energy Vehicle Technologies Office, as part of the Energy Efficient Mobility Systems program's Core Tools activity area. The views expressed herein do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting this work for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

Author Contributions

The authors confirm contribution to the paper as follows: algorithm development and implementation: G. Kosmacher; data collection, initial visualizations and problem definition: K. Shankari; draft manuscript preparation: G. Kosmacher and K. Shankari. All authors reviewed the results and approved the final version of the manuscript.

Data Accessibility Statement

All the data used in this paper is available publicly as part of the MobilityNet dataset. http://github.com/MobilityNet

References

1. Environmental Protection Agency. *Inventory of U.S. Greenhouse Gas Emissions and Sinks: 1990–2019*. Complete Report EPA 430-R-21-005, Washington, DC, USA, 2021. URL https://www.epa.gov/sites/default/files/2021-04/documents/us-ghg-inventory-2021-main-text.pdf.
2. 1969 Nationwide Personal Transportation Survey, 1969. URL https://www.fhwa.dot.gov/ohim/1969/1969page.htm.
3. Adella Santos, Nancy McGuckin, Hikari Yuriko Nakamoto, Danielle Gray, and Susan Liss. *Summary of Travel Trends:*

2009 National Household Travel Survey. Tech. Rep. FHWA-PL-11-022, Federal Highway Administration, 2011. URL http://nhts.ornl.gov/2009/pub/stt.pdf.

4. Stopher, P. R., P. Bullock, and F. Horst. *Exploring the Use of Passive GPS Devices to Measure Travel*. 2002, pp. 959–967. doi:10.1061/40632(245)120. URL https://ascelibrary.org/doi/abs/10.1061/40632%28245%29120.
5. Zhou, J. Sustainable commute in a car-dominant city: Factors affecting alternative mode choices among university students. *Transportation Research Part A: Policy and Practice*, Vol. 46, No. 7, 2012, pp. 1013–1029. doi:10.1016/j.tra.2012.04.001. URL http://linkinghub.elsevier.com/retrieve/pii/S0965856412000651.
6. Giaimo, G., R. Anderson, L. Wargelin, and P. Stopher. Will It Work?: Pilot Results from First Large-Scale Global Positioning System-Based Household Travel Survey in the United States. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2176, No. -1, 2010, pp. 26–34. doi:10.3141/2176-03. URL http://trb.metapress.com/openurl.asp?genre=article&id=doi:10.3141/2176-03.
7. Patterson, Z. and K. Fitzsimmon. DATAMOBILE: A SMARTPHONE TRAVEL SURVEY EXPERIMENT. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2594, No. 1, 2016, pp. 35–43. doi:https://doi.org/10.3141/2594-07.
8. Hongik University, J. S. Lee, P. C. Zegras, F. Zhao, D. Kim, and J. Kang. Testing the Reliability of a Smartphone-Based Travel Survey: An Experiment in Seoul. *The Journal of The Korea Institute of Intelligent Transport Systems*, Vol. 15, No. 2, 2016, pp. 50–62. doi:10.12815/kits.2016.15.2.050. URL http://journal.kits.or.kr/journal/article.php?code=39731.
9. Litman, T. *Short and Sweet: Analysis of Shorter Trips Using National Personal Travel Survey Data*. Tech. rep., Victoria Transport Policy Institute, 2012. URL http://www.vtpi.org/short_sweet.pdf.
10. Board, T. R. and National Academies of Sciences, Engineering, and Medicine. *Activity-Based Travel Demand Models: A Primer*. The National Academies Press, Washington, DC, 2014. doi:10.17226/22357. URL https://nap.nationalacademies.org/catalog/22357/activity-based-travel-demand-models-a-primer.
11. McNally, M. G. and C. R. Rindt. The Activity-Based Approach. In *Handbook of Transport Modelling*, Vol. 1 (D. A. Hensher and K. J. Button, eds.). Emerald Group Publishing Limited, 2007, pp. 55–73. doi:10.1108/9780857245670-004. URL https://doi.org/10.1108/9780857245670-004.
12. McNally, M. G. The Four-Step Model. In *Handbook of Transport Modelling*, Vol. 1 (D. A. Hensher and K. J. Button, eds.). Emerald Group Publishing Limited, 2007, pp. 35–53. doi:10.1108/9780857245670-003. URL https://doi.org/10.1108/9780857245670-003.
13. Shankari, K., L. Boyce, E. Hintz, and A. Duvall. *The CanBikeCO Mini Pilot: Preliminary Results and Lessons Learned*. Tech. rep., NREL, 2021.
14. rMove. https://rmove.rsginc.com/, 2023. Accessed: 2023-04-05.
15. Westat DailyTravel. https://apps.apple.com/us/app/westat-dailytravel/id1152627256, https://play.google.com/store/apps/details?id=trip_builder_mobile.trip_builder_mobile&gl=US, 2023. Accessed: 2023-04-05.
16. Vij, A. and K. Shankari. When is big data big enough? Implications of using GPS-based surveys for travel demand analysis. *Transportation research. Part C, Emerging technologies*, Vol. 56, 2015, pp. 446–462.
17. Prelipcean, A. C., G. Gidofalvi, and Y. O. Susilo. Measures of transport mode segmentation of trajectories. *International Journal of Geographical Information Science*, Vol. 30, No. 9, 2016, pp. 1763–1784. doi:10.1080/13658816.2015.1137297. URL https://doi.org/10.1080/13658816.2015.1137297. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/13658816.2015.1137297.
18. Android Doze Documentation. https://developer.android.com/training/monitoring-device-state/doze-standby, 2022. Accessed: 2022-06-12.
19. Apple Configuring background execution modes Documentation. https://developer.apple.com/documentation/xcode/configuring-background-execution-modes, 2022. Accessed: 2022-06-12.
20. Shankari, K. *e-mission: an open source, extensible platform for human mobility systems*. Ph.D. thesis, EECS Department, University of California, Berkeley, 2019. URL http://www2.eecs.berkeley.edu/Pubs/TechRpts/2019/EECS-2019-180.html.
21. Patterson, Z., K. Fitzsimmons, S. Jackson, and T. Mukai. Itinerum: The Open Smartphone Travel Survey Platform. *SoftwareX*, Vol. 10, 2019, p. 100230. doi:10.1016/j.softx.2019.04.002. URL https://linkinghub.elsevier.com/retrieve/pii/S2352711018300980.
22. Allen, J. Maintaining knowledge about temporal intervals. *Commun. ACM; (United States)*, Vol. 26, No. 11, 1983, pp. 832–843.
23. Zandbergen, P. A. Accuracy of iPhone Locations: A Comparison of Assisted GPS, WiFi and Cellular Positioning. *Transactions in GIS*, Vol. 13, No. s1, 2009, pp. 5–25. doi:10.1111/j.1467-9671.2009.01152.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9671.2009.01152.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9671.2009.01152.x.
24. Apple CLLocationManager Class Documentation. https://developer.apple.com/documentation/

corelocation/cllocationmanager?language=objc, 2022. Accessed: 2022-06-12.

25. Kjærgaard, M. B., H. Blunck, T. Godsk, T. Toftkjær, D. L. Christensen, and K. Grønbæk. Indoor Positioning Using GPS Revisited. In *Pervasive Computing* (P. Floréen, A. Krüger, and M. Spasojevic, eds.). Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2010, pp. 38–56. doi:10.1007/978-3-642-12654-3_3.
26. Ranasinghe, C. and C. Kray. Location Information Quality: A Review. *Sensors*, Vol. 18, No. 11, 2018, p. 3999. doi:10.3390/s18113999. URL https://www.mdpi.com/1424-8220/18/11/3999. Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.
27. Blunck, H., M. B. Kjærgaard, and T. S. Toftegaard. Sensing and Classifying Impairments of GPS Reception on Mobile Devices. In *Pervasive Computing* (K. Lyons, J. Hightower, and E. M. Huang, eds.). Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2011, pp. 350–367. doi:10.1007/978-3-642-21726-5_22.
28. Ranacher, P., R. Brunauer, W. Trutschnig, S. Van der Spek, and S. Reich. Why GPS makes distances bigger than they are. *International journal of geographical information science : IJGIS*, Vol. 30, No. 2, 2016, pp. 316–333.
29. Bähr, S., G.-C. Haas, F. Keusch, F. Kreuter, and M. Trappmann. Missing Data and Other Measurement Quality Issues in Mobile Geolocation Sensor Data. *Social Science Computer Review*, Vol. 40, No. 1, 2022, pp. 212–235. doi:10.1177/0894439320944118. URL https://doi.org/10.1177/0894439320944118. _eprint: https://doi.org/10.1177/0894439320944118.
30. em-public-dashboard Repository. https://github.com/e-mission/em-public-dashboard/blob/main/viz_outputs/Program_trip_characteristics.ipynb, 2023. Accessed: 2023-01-01.
31. Reddy, S., M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. Using mobile phones to determine transportation modes. *ACM transactions on sensor networks*, Vol. 6, No. 2, 2010, pp. 1–27.
32. Zheng, Y., Y. Chen, Q. Li, X. Xie, and W.-Y. Ma. Understanding transportation modes based on GPS data for web applications. *ACM transactions on the web*, Vol. 4, No. 1, 2010, pp. 1–36.
33. Zhong, M., J. Wen, P. Hu, and J. Indulska. Advancing Android activity recognition service with Markov smoother: Practical solutions. *Pervasive and mobile computing*, Vol. 38, 2017, pp. 60–76.
34. Shah, R., C.-y. Wan, H. Lu, and L. Nachman. Classifying the mode of transportation on mobile phones using GIS information. In *Proceedings of the 2014 ACM International Joint Conference on pervasive and ubiquitous computing*. UbiComp '14, ACM, 2014, pp. 225–229.
35. Yang, F., Z. Yao, and P. J. Jin. GPS and Acceleration Data in Multimode Trip Data Recognition Based on Wavelet Transform Modulus Maximum Algorithm. *Transportation research record*, Vol. 2526, No. 1, 2015, pp. 90–98.
36. Gonzalez, P., J. Weinstein, S. Barbeau, M. Labrador, P. Winters, N. Georggi, and R. Perez. Automating mode detection for travel behaviour analysis by using global positioning systems-enabled mobile phones and neural networks. *IET intelligent transport systems*, Vol. 4, No. 1, 2010, pp. 37–37.
37. Prelipcean, A. C., G. Gidófalvi, and Y. O. Susilo. Mobility Collector. *Journal of Location Based Services*, Vol. 8, No. 4, 2014, pp. 229–255. doi:10.1080/17489725.2014.973917. URL https://doi.org/10.1080/17489725.2014.973917.
38. MobilityNet Repository. https://github.com/MobilityNet, 2022. Accessed: 2022-06-12.
39. Shankari, K., J. Fuerst, M. Fadel Argerich, E. Avramidis, and J. Zhang. MobilityNet: Towards a Public Dataset for Multimodal Mobility Research. *Climate Change AI 2020 @ ICLR20 (Spotlight Talk)*. URL https://github.com/MobilityNet/mobilitynet.github.io.
40. Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006, p. 38.
41. e-mission Repository. https://github.com/e-mission, 2022. Accessed: 2022-06-12.
42. Shankari, K., M. Yin, D. Culler, and R. Katz. E-mission: Automated transportation emission calculation using smartphones. In *2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*. 2015, pp. 268–271. doi:10.1109/PERCOMW.2015.7134044.
43. Scikit Learn Confusion Matrix Documentation. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html, 2022. Accessed: 2022-06-12.
44. Shankari, K., M. Yin, S. Shanmugam, D. E. Culler, and R. H. Katz. *E-Mission: Automated transportation emission calculation using smart phones*. Tech. Rep. UCB/EECS-2014-140, EECS Department, University of California, Berkeley, 2014. URL http://www2.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-140.html.
45. openstreetmap Overpass API Documentation. https://wiki.openstreetmap.org/wiki/Overpass_API, 2022. Accessed: 2022-06-12.