I Know I'm Right, But Does my Phone?
Evaluating the Interplay between Trajectory Segmentation and Mode Inference Errors

Gabriel Kosmacher

Office of Science, Science Undergraduate Laboratory Internship Program

University of Illinois Urbana-Champaign, Champaign, IL

National Renewable Energy Laboratory

Golden, Colorado

August 10, 2022

Prepared in partial fulfillment of the requirement of the Department of Energy, Office of Science's Science Undergraduate Laboratory Internship Program under the direction of K. Shankari at the National Renewable Energy Laboratory. This work was submitted for publication to the *Transportation Research Record: Journal of the Transportation Research Board* on August 1, 2022.

Participant: _____
Gabriel Kosmacher

Research Advisor: _____
K. Shankari

ABSTRACT

Transportation is the largest source of green-house gas emissions in the United States. Reducing transportation emissions depends on human travel behavior, which relies on local land use and planning. NREL OpenPATH is an open source, extensible platform that tackles this challenge through smartphone-based travel diary creation. 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 GPS tracking surveys. Mode inference algorithms in particular have been well represented in literature. However, these algorithms have typically been validated using prompted recall of pre-segmented trips, which doesn't account for segmentation error, thus disregarding the *continuity* of mode inference. Furthermore, phone operating systems and applications have adopted battery-conserving techniques, but we are not aware of prior work that has characterized the resulting data collection errors or evaluated procedures to mitigate them.

We introduce a framework to evaluate accuracy of trip length computations and mode inference. We develop a temporal alignment procedure in analyzing continuous mode-segmented trajectories for groups of trips. We then *apply* our framework to the OpenPATH pipeline using MobilityNet, a public dataset containing information from three *artificial timelines* that cover 15 different travel modes. Our results show that inference based on an integration with map features results in weighted F_1 scores of 0.60 (iOS) and 0.74 (android). We also show that OpenPATH tends to under count trip length (android: mean: -0.0438, variance: 0.00696, iOS: mean: -0.0704, variance: 0.065).

Keywords: NREL OpenPATH, Human Mobility Tracking, Trajectory Measurement Error, Transportation Mode Segmentation, Interval Algebra

INTRODUCTION

Transportation is the largest source of green-house gas emissions in the United States (*1*). Reducing transportation emissions depends on human travel behavior, which relies on strategic local land use and planning. **NREL OpenPATH** (née e-mission) 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 1, where H is a history of trips T with length $T.\ell$, trips consist of unimodal sections S^m with length $\alpha_{S^m} T.\ell$, and each mode m has a corresponding energy impact E.

$$\sum_{T \in H} \sum_{S^m \in T} E_{S^m} \times \alpha_{S^m} T.\ell \tag{1}$$

OpenPATH utilizes a **sensor-based multi-step pipeline**[1] for processing phone sensor data and outputting section segmentation and mode inference. The pipeline is a sequence of independent deterministic stages (Fig. 2) so that the input to one stage is the output of another. The only permanent stage is the data received from the OpenPATH smartphone app, which takes advantage of a particular phone's built in, black box, virtual sensors.

OpenPATH travel diaries have various use cases. Shankari et al. (*2*) uses OpenPATH travel diaries and energy calculation in direct evaluation of programs aimed at shifting behavior. Activity based models (*3, 4*) or four-step models (*5*) can use OpenPATH travel diaries at the regional level to assess the impact of future changes and prioritize infrastructure.

However, the information gleaned from OpenPATH is only as accurate as the underlying methods used by the platform, and while there has been much work implementing systems to instrument travel behavior, there exists no consensus procedure of evaluation. Motivating this phenomena is that many such systems exist in the personal, not social, domain. In the personal domain, users make decisions based on self-tracking and have an intuition of accuracy based on experienced ground truth. The decisions are low-stakes lifestyle changes, which may have individual meaning but have 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 accuracy of the data before making its decisions (*6*).

Despite the fact that there exists no 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 framework to evaluate multi-step sensor-based pipelines. 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.

[1] https://github.com/e-mission

mode inference algorithms Validation of mode inference has typically used prompted recall of pre-segmented trips, or uses edge buffering to match segments, neither of witch account for segmentation error, thus disregarding the *continuity* of mode inference.

Furthermore, 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[2]. iOS has always restricted background app operation unless actively using certain features such as location[3]. And smartphone apps can choose to reduce accuracy or frequency of data collection when not in active use (*7, 8*). However, we are not aware of prior work that characterized the resulting data collection errors or evaluated procedures to mitigate them.

We introduce a framework to evaluate accuracy of trip length computations and mode inference. To determine accuracy of trip length, the framework 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 Prelipcean et al. (*9*), itself a variation of time algebra alignment proposed by Allen (*10*).

The framework is applied to the OpenPATH pipeline using **MobilityNet**[4], a public dataset containing information from three *artificial timelines* that cover 15 different travel modes (*11*). 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). In the following sections, we denote both spatial and temporal ground truth simply as *ground truth*. The use of artificial timelines allow for repetition of data collection and serves as control data, baseline requirements for any scientific experiment.

Shankari et al. (*11*) (Ch. 7.2) uses MobilityNet to evaluate power drain on various evaluation phones. Rigorous analysis shows that power consumption on android is primarily affected by data collection frequency and by data collection accuracy on iOS. As a result of this analysis, OpenPATH currently uses high accuracy and medium frequency settings for data collection on android phones, and high accuracy and high frequency settings on phones running iOS. High accuracy settings tends to favor GPS for collecting location data, while high frequency settings will sense and processes data more often.

We evaluate the OpenPATH pipeline at different stages and on phones with varying sensor configurations. Our results show that 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). We also show that OpenPATH tends to under count trip length, our analysis yielding the following results for signed relative error (android: mean: -0.0438, variance: 0.00696, iOS: mean: -0.0704, variance: 0.065).

[2]https://developer.android.com/training/monitoring-device-state/doze-standby

[3]https://developer.apple.com/documentation/xcode/configuring-background-execution-modes

[4]https://github.com/MobilityNet

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 framework. Then, we perform an empirical evaluation of the OpenPATH pipeline on the MobilityNet dataset. The penultimate section is a discussion on our framework, empirical results, and sources of error. Finally, we conclude and discusses 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

OpenPATH collects location data in the model of a *passive travel survey*. It uses notoriously unpredictable (*12*) *fused* location data – modern iPhones do not allow access to GPS directly[5] – and thus errors in these geospatial points are inherent to trip length error. Researchers have shown many factors that affect goelocation collection on smartphones; from indoor locations (*13*) to underlying APIs (*12, 14*) to how a user physically handles the phone (*15*). These errors exist in conjunction with the systematically overestimated distance by GPS as shown in Ranacher et al. (*16*).

However, none of these studies provide insight into how these error propagate thought a passive data collection run to effect total trip length measurements. We expand our search to errors in other passively collected travel surveys, where there has been considerably less work. Bähr et al. (*17*) proposes a method to address missingness in geolocation data collected by smartphones. Missingness is an important aspect of analysing aggregate trip survey behavior. Yet geolocation missingness does not describe trip length error as a whole, having nothing to do with the error introduced by trip segmentation or location resampling algorithms.

Mode inference

OpenPATH assigns modes to *continuous mode-segmented trajectories*, and trajectory segmentation error is inherent to mode inference error. Researchers have used decision trees (*18, 19*), Hidden Markov Models (*20, 21*), and neural networks (*22, 23*) to distinguish between travel modes. However, the algorithms either process unimodal trips (*18, 20, 21, 23*), or evaluate mode inference and section segmentation separately (*19, 22*).

Prelipcean et al. (*9*) introduce a framework to evaluate continuous transportation mode segmentation by aligning temporal intervals, motivated by the work of Allen (*10*). This work serves as a baseline for our framework of continuous transportation mode segmentation, but out temporal alignment differs in a few key ways. Namely, we do not take into account spatial penalties and segmentation count penalties as our mode prediction segmentation is solely based on time delineation.

[5] https://developer.apple.com/documentation/corelocation/cllocationmanager?language=objc

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 Prelipcean et al., Prelipcean et al. (*9, 24*) to be consistent with the formalisms of SHANKARI (*7*). In particular, we add formalisms and definitions for *base modes*, *segmentation decomposition*, *histories*, and *length measures*, while renaming *tripleg* as *section*.

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 $b : M \rightarrow BM$ 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 2. *Section length* is defined to be the sum distance between consecutive locations as in Equation 3, where $||\cdot||$ is the euclidean norm.

$$S^m = \langle l_i^{x,y,t,m} : l_{i+1}.t > l_i.t \wedge l_i.t \in [s; e] \wedge l_i.m \in M \wedge l_i.m = l_{i+1}.m \rangle \tag{2}$$

$$S^m.\ell = \sum_{i}^{n-1} ||l_i - l_{i+1}|| \tag{3}$$

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 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 4. W_i^{i+1} represents the transition period between segments 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 algorithms given *dwell time buffer*. *Trip length* is defined to be the sum section length, as in Equation 5.

$$T = \langle S_i^{m_i},\ W_i^{i+1} : T.s = S_0.s \wedge T.e = S_n.e \wedge W_i^{i+1}.s = S_i.e \wedge W_i^{i+1}.e = S_{i+1}.s \wedge S_i.e \leq S_{i+s}.s \rangle \tag{4}$$

$$T.\ell = \sum_{i}^{|T|} S_i^m.\ell \tag{5}$$

Mode-segmented trajectories and histories

A *mode-segmented trajectory* is a sequence of trips in temporal order, with strictly non-zero dwell times between trips, where the start of the trajectory is the start of the first trip, 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 6. 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 trips are replaced by their ordered constituent sections, as in Equation 7. *Mode-segmented trajectory length* is defined to be the sum section length, as in Equation 8.

$$L = \langle T_i,\ WT_i^{i+1} : T_i.e < T_{i+1}.s \wedge WT_i^{i+1}.s = T_i.e \wedge WT_i^{i+1}.e = T_{i+1}.s \rangle \tag{6}$$

$$LS = \langle S_{i,j}^{m_i}\ W_{i,j}^{i+1,j},\ WT_j^{j+1} : S_{i,j}^{m_i} = S_j^{m_i},\ W_{i,j}^{i+1,j} \in T_i, \wedge T_i \in L \wedge WT_j^{j+1} \in L \rangle \tag{7}$$

$$L.\ell = \sum_{i}^{|L|} T_i.\ell \tag{8}$$

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 union between the segment decomposition of a mode-segmented trajectory L and the ordered sequence of each WT_i^{i+1}, as shown in Equation 9. *History length*, $H.\ell$, equals the associated mode-segmented trajectory length $L.\ell$.

$$H = \langle LS_i,\ WT_i^{i+1} : LS_i.s < LS_{i+1}.e \wedge WT_i^{i+1}.s = LS_i.e \wedge WT_i^{i+1}.e = LS_{i+1}.s \rangle \tag{9}$$

EVALUATION FRAMEWORK

This section presents a framework to measure trip length and mode inference, revolving around the comparison of ground truth histories H^{GT} and inferred histories H^{inf}.

Trip length

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 then taking the subway to the ballgame as separate walking and subways 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 linear combination of trip lengths (Eq. 8). 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 a error *distribution* does provide valuable information. Mean signed relative error for a set of histories can expose trends in history length bias, while 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: $H_m^{GT} \neq \emptyset$

if $H_m^{inf} = \emptyset$ **then** ▷ no inferred history
  SE $\leftarrow -H_m^{GT}.\ell$ ▷ signed error is the ground truth length
else
  SE $\leftarrow H_m^{inf}.\ell - H^{GT}.\ell$ ▷ signed error
end if
SRE $\leftarrow$ SE $/H_m^{GT}.\ell$ ▷ signed error relative to ground truth length
return SE, SRE

Mode inference

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 Prelipcean et al. (*9*) Section 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 Prelipcean et al. (*9*) in that we temporally align *solely* based on interval overlap as defined in Allen (*10*). Investigating temporal overlap allows for evaluation of *how often* a mode was correctly inferred, in accordance with the motivations of traditional multinomial classification problems (*25*).

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).

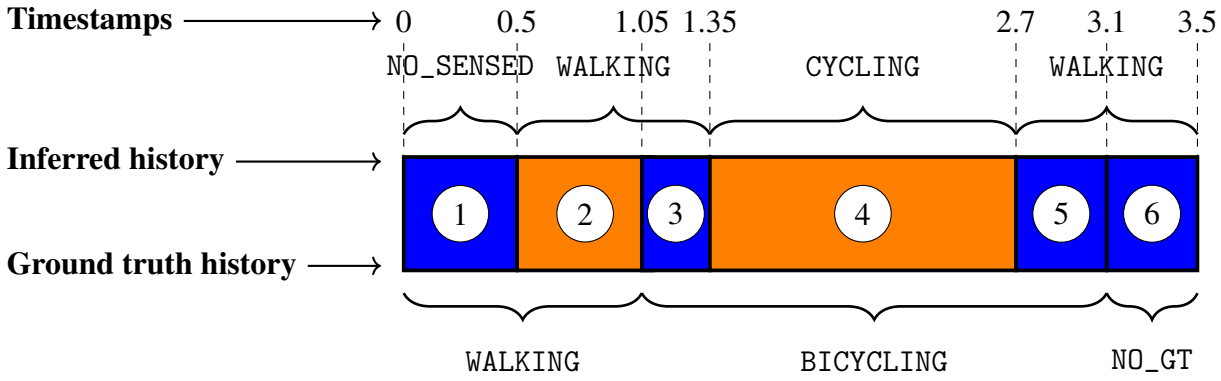

FIGURE 1 Temporal history alignment for inferred and ground truth histories: (1) we begin walking but data collection has not started, (2) we are walking and pipeline correctly infers walking, (3) we begin biking but pipeline still infers walking, (4) we are biking, which the pipeline correctly infers,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

Consider Figure 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 where walking and *when* we are 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 Figure 1, time ranges (2) and (4) represent modal overlap, while (1), (3), (5), and (6) are the time ranges where there is no overlap and an error has occurred.

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 sub-sequence of aligned timestamps (e.g. elements 2,3,4 in Figure 4. 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. At the end of the alignment process, we have histories that are an ordered sequence of temporally aligned uni-modal *elements*.

The pseudocode in Algorithm 2 details the temporal alignment process with respects 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 history 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 $|H_a^{inf}| \leq |H^{inf}| + |H^{GT}|$.

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
      $H_a^{inf}.\text{append}(h_a : h_a.m = h_i^{inf}.m)$ ▷ append to inf with inf mode
      $H_a^{GT}.\text{append}(h_a : h_a.m = h_j^{GT}.m)$ ▷ 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}

EMPIRICAL EVALUATIONS

This section motivates and develops metrics to apply our evaluation framework and evaluate the performance of the NREL OpenPATH pipeline (Fig. 2) using *artificial timelines* from the Mobility-Net dataset.

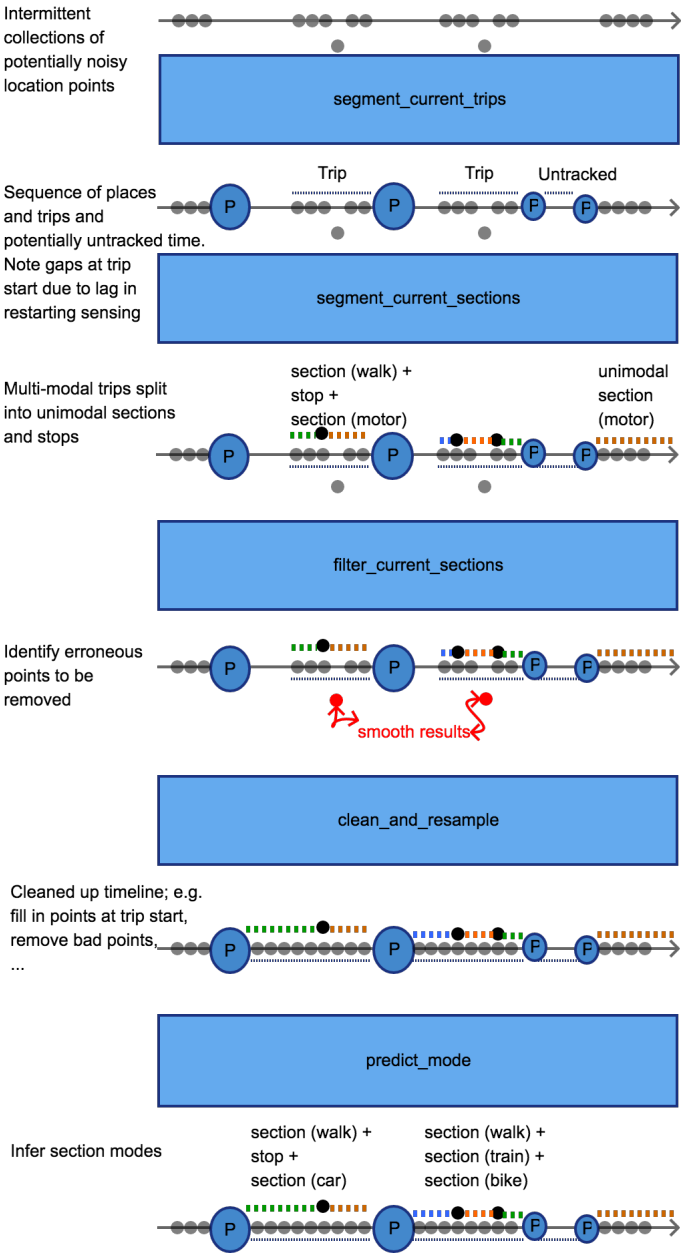

FIGURE 2 Sensor-based multi-step OpenPATH pipeline as seen in SHANKARI (*7*) Figure 5.1**.**

The MobilityNet dataset contains 1080h of multi-modal travel from 3 artificial timelines (*11*). 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. High accuracy settings tend to favor GPS for collecting location data, while high frequency settings will sense and processes data more often. Duty cycling allows for high accuracy and high frequency settings, but has sensing gaps at trip start and end due to sensing delay.

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*.

Trip length

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

Performance analysis

We evaluate trip length at the `segment_current_trip` (raw) and `clean_and_resample` (clean) NREL OpenPATH pipeline stages.

Table 1 has the sample mean, variance, min, max, and size of the signed relative error distribution at selected sensor settings. We also notice that both clean phones have a small variance and negative sample mean at the clean output. From this, we can say with confidence that there is a systematic underestimation of trip length at the clean output stage.

			$\bar{x}$	s^2	min	max	n
raw	android	`HAMFDC`	0.197	0.217	-0.321	2.183	42
	ios	`HAHFDC`	-0.0728	0.0559	-1.0	0.419	30
clean	android	`HAMFDC`	-0.0438	0.00696	-0.363	0.0812	32
	ios	`HAHFDC`	-0.0704	0.065	-1.0	0.001578	42

TABLE 1 Statistics for signed relative error trip length distribution

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 stage as opposed to raw. 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. For further analysis and examples of errors resulting in overestimation and underestimation error, see our discussion on trip length in the penultimate section.

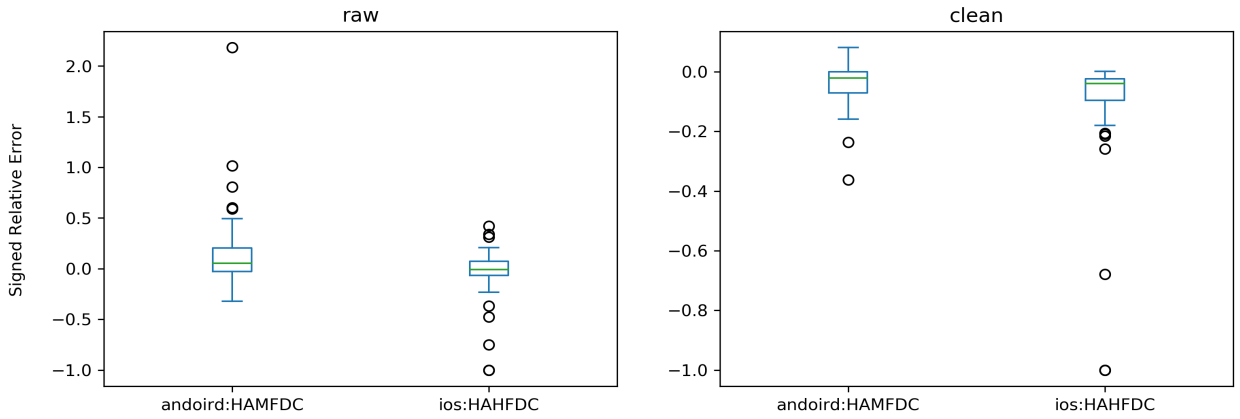

FIGURE 3 Signed Relative Error Box Plots

Mode inference

We evaluate the mode inference error as a multinomial classification problem, building confusion matrices to see what mistakes our classifier makes and calculating F_1 scores to measure our classifiers accuracy, both for individual base modes and in aggregate.

Confusion matrix

The construction of confusion matrices for discrete classification problems is well defined and implemented[6]. We adapt this methodology to our continuous problem, recording duration of mode overlap in 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 a inferred mode (line 1) and each entry represents the total time spent in a aligned ground truth mode (lines 3-8), where M^{GT} is a set of *ground truth modes and* and M^{inf} is a set of *inferred modes*.

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

[6]https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Multinomial classification

Temporal alignment allows for the construction of the *continuous* multinomial classification process, which is detailed in relation to the pseudocode in 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 us to compare histories of different modal sets. For instance, a set of ground truth modes might be {WALKING, BICYCLING, E_SCOOTER} while the set of inferred modes in {WALKING, CYCLING}. In this scenario, the base mode map would take {CYCLING, BICYCLING, E_SCOOTER} to {CYCLING}, the associated base mode. Classification is performed via a one-verse-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. classification_metrics(·) must first align the histories (line 1), after which it begins cycling through a set of base modes (line 2 & 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.

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

Require: *ordered* H^{inf}, H^{GT} **and** $H^{GT} \neq \emptyset$

1: H_a^{inf}, H_a^{GT} = temporal_align(H^{inf}, H^{GT})
2: $BM = \{b(m) : m \in M^{inf} \cup M^{GT}$ ▷ construct base mode set
3: **for** $bm \in BM\}$ **do** ▷ loop over base modes
4: **for** $i \leftarrow 0, \ldots, |H_a|$ **do**
5: duration $= h_{a_i}.e - h_{a_i}.s$
6: **if** $bm = b(h_{a_i}^{inf}.m)$ and $bm = b(h_{a_i}^{GT}.m)$ **then** ▷ inf and GT in basemode
7: $TP^{bm} \leftarrow TP^{bm} +$ duration ▷ true positive
8: **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
9: $FP^{bm} \leftarrow FP^{bm} +$ duration ▷ False positive
10: **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
11: $FN^{bm} \leftarrow FN^{bm} +$ duration ▷ False negative
12: **else**
13: $TN^{m} \leftarrow TN^{m} +$ duration ▷ True negative
14: **end if**
15: **end for**
16: **end for**
17: **return** $\left\{\left(TP^{bm},\ FP^{bm},\ FN^{bm},\ TN^{bm}\right) : bm \in \{b(m) : m \in M^{inf} \cup M^{GT}\}\right\}$

Outputs of the classification_metrics(·)-function can be used to calculate the F_1 *score* of a base mode, as shown in Equation 10.

$$F_1^{bm} = \frac{2 \cdot TP^{bm}}{2 \cdot TP^{bm} + FN^{bm} + FP^{bm}} \tag{10}$$

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 11.

$$F_1^{avg} = \sum_{i=1}^{|BM|} \left[\sum_{j=1}^{|M_{bm_i}|} \sum_{k=1}^{|M^{inf}|} cm_{j,k} \right] \cdot F_1^{bm} \tag{11}$$

Performance analysis

The NREL OpenPATH pipeline currently supports 4 methods of mode prediction

- `segmentation/raw_trip, segmentation/raw_section` (Raw)
- `analysis/cleaned_trip, analysis/cleaned_section` (Clean)
- `analysis/cleaned_trip, analysis/inferred_section` (Inferred)
  - Random forest
  - rule+GIS based

Raw mode prediction results from activity detection APIs on both android and iOS that can distinguish between active modes and a single motorized mode. Currently, the android API distinguishes between `WALKING, CYCLING, AUTOMOTIVE` while the iOS API can distinguish between `WALKING, RUNNING, CYCLING, AUTOMOTIVE`.

Clean mode prediction happens *after* the `clean_and_resample` pipeline stage that removes extraneous points and fills in missing points at the start of a trip. Clean mode prediction uses the same activity detection API as in raw mode inference, while also inferring the mode `AIR_OR_HSR` for particularly long and fast trips.

Random forest based mode inference is the traditional option, with a feature set largely based on Zheng et al. (*19*) and detailed in Shankari et al. (*26*) and Shankari et al. (*27*). The model distinguishes between 6 modes - `WALKING, BICYCLING, BUS, CAR, TRAIN, AIR_OR_HSR`. It requires a large reference dataset for training, and once trained 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 non-motorized modes based off speed. Motorized modes are inferred by querying OpenStreetMap (OSM) using the Overpass API[7] to determine weather endpoints 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, BICYCLING, BUS, CAR, TRAIN, AIR_OR_HSR`.

Figure 4 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. 11), in which GIS

[7] https://wiki.openstreetmap.org/wiki/Overpass_API

based mode inference scores 0.71 and 0.61 for android and iOS selected configuration settings, while random forest based mode inference scores 0.28 and 0.21.

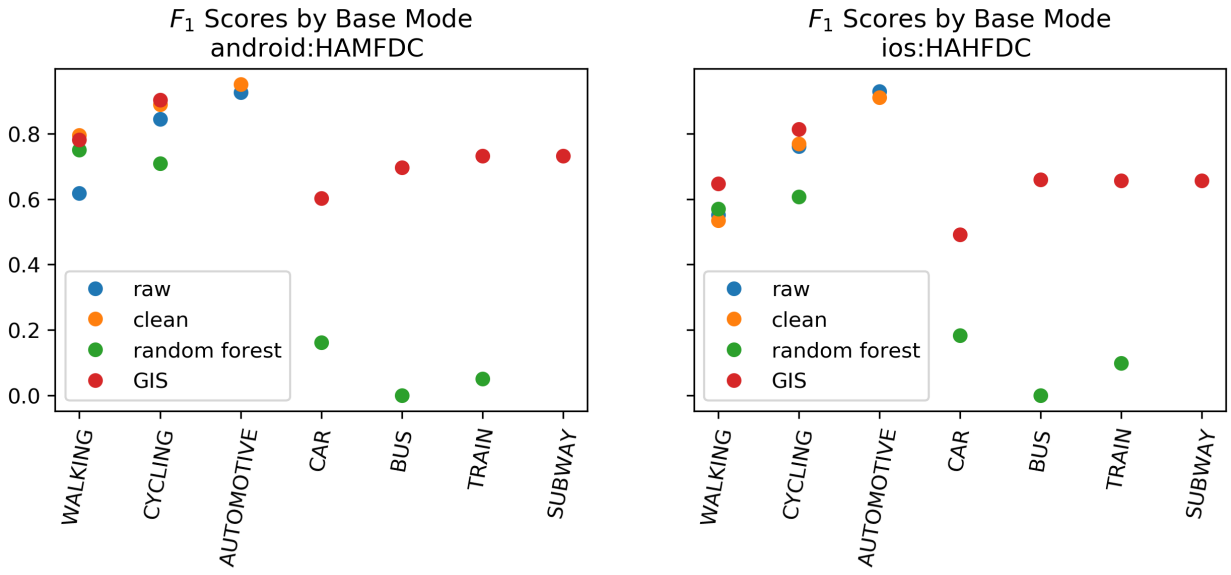

FIGURE 4 F_1 Scores by Base Mode for Selected OS Setting Configurations

Figures 5, 6 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 towards predicting a train trip as a car ride. Figure 11 in Shankari et al. (*27*) can help explain this phenomena, displaying the support of the random forest training set, with an noticeable overrepresentation of `CAR` for the `AUTOMOTIVE` base mode.

Not surprisingly, the bias towards `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 do not sense any movement during the walking portion of our trip for clean, random forest, and GIS output stages 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 raw output stage (Fig. 5, 6). 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 there is no ground truth in the middle of a trip. This suggests that iOS has trouble discerning the transition between walking and stopping during a trip.

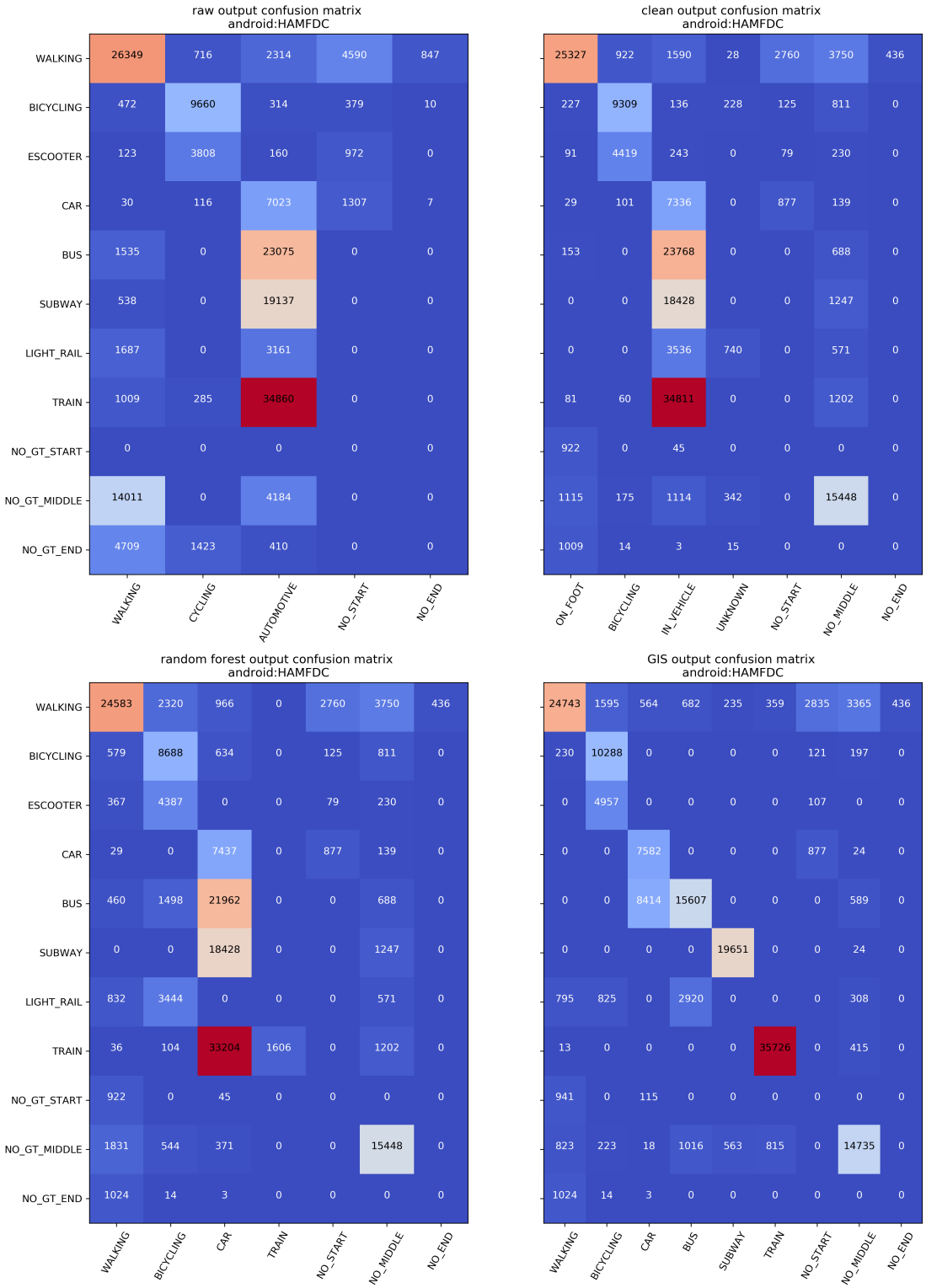

FIGURE 5 Confusion Matrices at each Pipeline Output Stage on Phones Running Android for Selected Setting Configurations

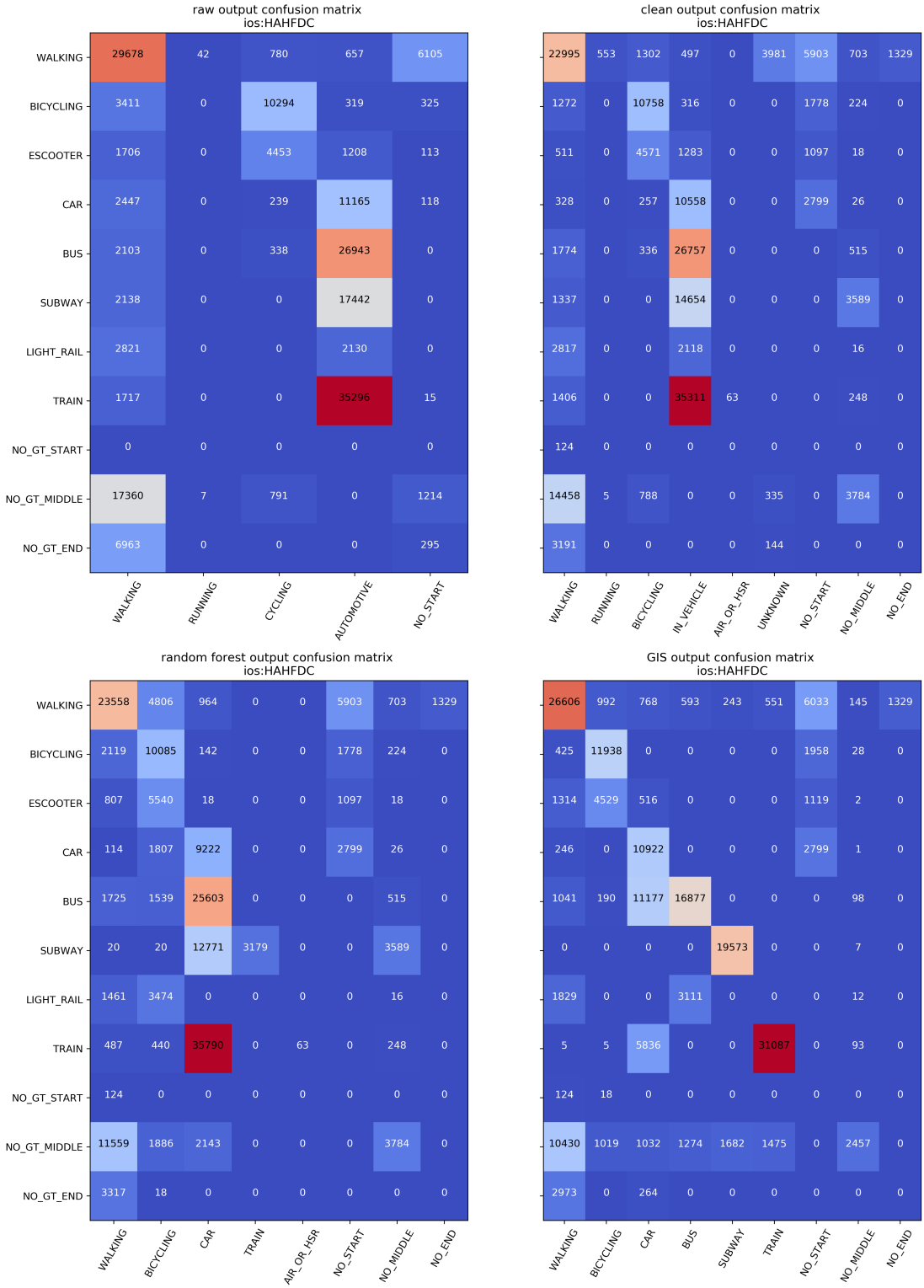

FIGURE 6 Confusion Matrices at each Pipeline Output Stage on Phones Running iOS for Selected Setting Configurations

DISCUSSION

This paper present a framework that allows for informative cross-sectional and longitudinal studies of arbitrary multi-step sensor-based pipelines. Trip length error is analyzed via standard methods for measurement error, taking care to avoid double counting from poorly segmented trips. Mode inference accuracy is measured through novel temporal alignment, allowing for mode inference to be evaluated as a continuous multinomial classification problem. These methods are applied on MobilityNet, providing valuable information about the NREL OpenPATH pipeline.

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 towards underestimation with little influence from random error.

Evaluations of mode inference show that rule+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 has poor allocation of support, as previously discussed, and relies on recruiting a small number of collectors where ground truth is reported by prompted recall (Tab. 1 in Shankari et al. (*26*)).

For the remainder of this section, we discuss sources of trip length and mode inference error from the NREL OpenPATH pipeline as originally reported in SHANKARI (*7*).

Trip length

We detail through example the main causes of *overestimation* and *underestimation* of trip length on the NREL OpenPATH pipeline.

There are two main sources leading to overestimation of trip length for the OpenPATH pipeline, namely:

- The sensed points are spatially offset from the real trajectory (Fig. 7; Top). These errors will lead to overestimation of trip length as a ground truth distance of a straight line will always be shorter that the inferred distance including the offset point.
- The sensed points have temporal inconsistencies (Fig. 7; Bottom). These errors will lead to overestimation as inferred trip length will include repeatedly zig-zagging between the current point and an earlier one, with each repetition adding an additional distance to total trip length.

This comes in contrast with the two main sources of underestimation:

- The sensed points are collected at a delay (Fig. 8). These errors tend to cause a shorter time range for data collection, thus leading to a shorter trip length, all else being equal.
- The sensed points are under reported (Fig.9). These errors cause poor resolution of trip trajectory, leading to an underestimation of trip length as more of the trip is inferred to be traveled in a straight line, in opposition to ground truth with 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.

It is clear to see how over *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 OpenPATH has a systematic bias toward *under representing* trip length (Tab. 1), implying that our overestimation errors hold less impact in our total trip length calculations.

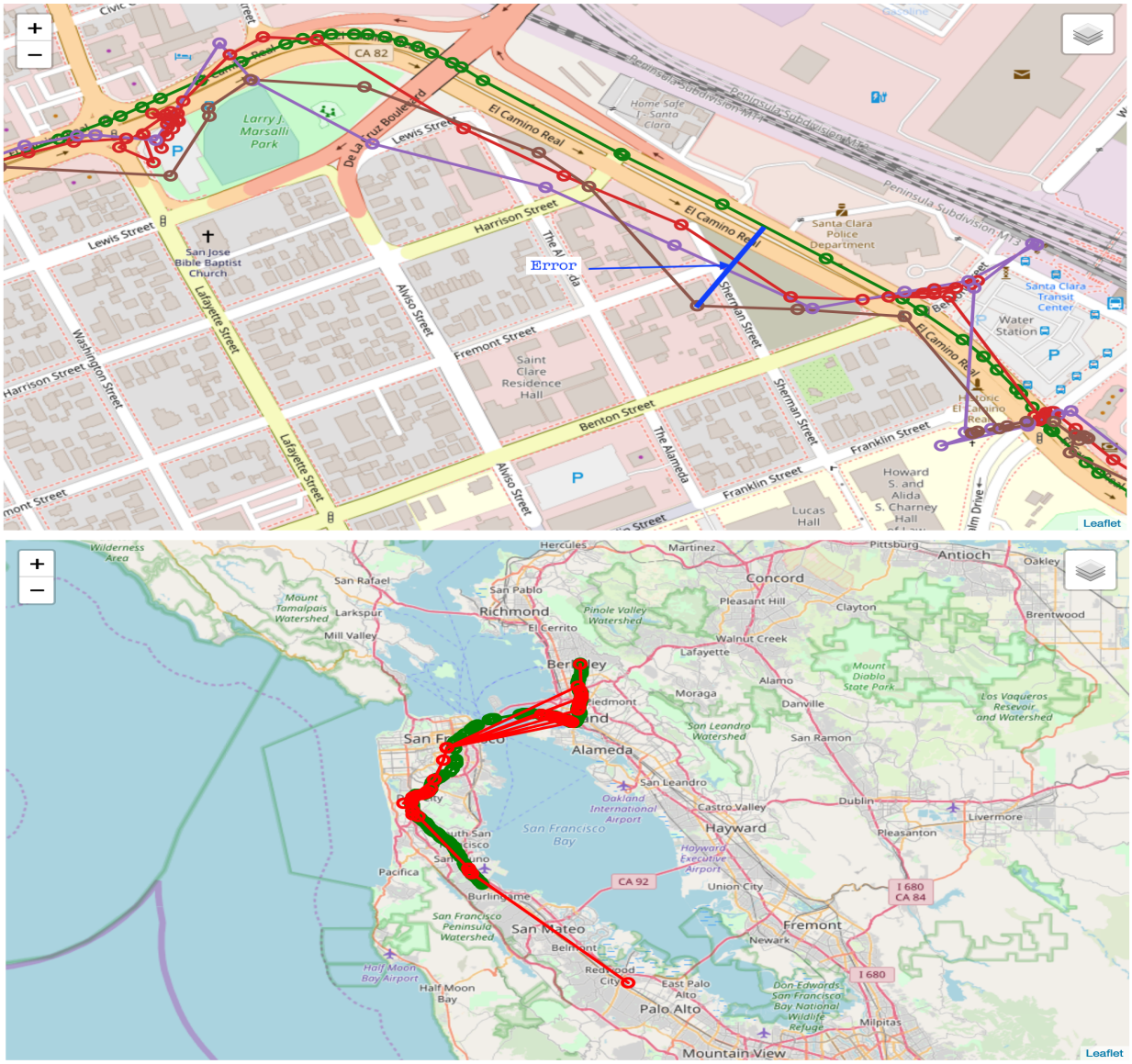

FIGURE 7 Examples of errors in trajectory.
Top – Spatial errors from iOS `MAHFDC` runs: The green line is ground truth, other colors being sensed data. The spatial error is the shortest perpendicular distance to the spatial ground truth line (i.e., the thick blue line from the brown point to the green 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.

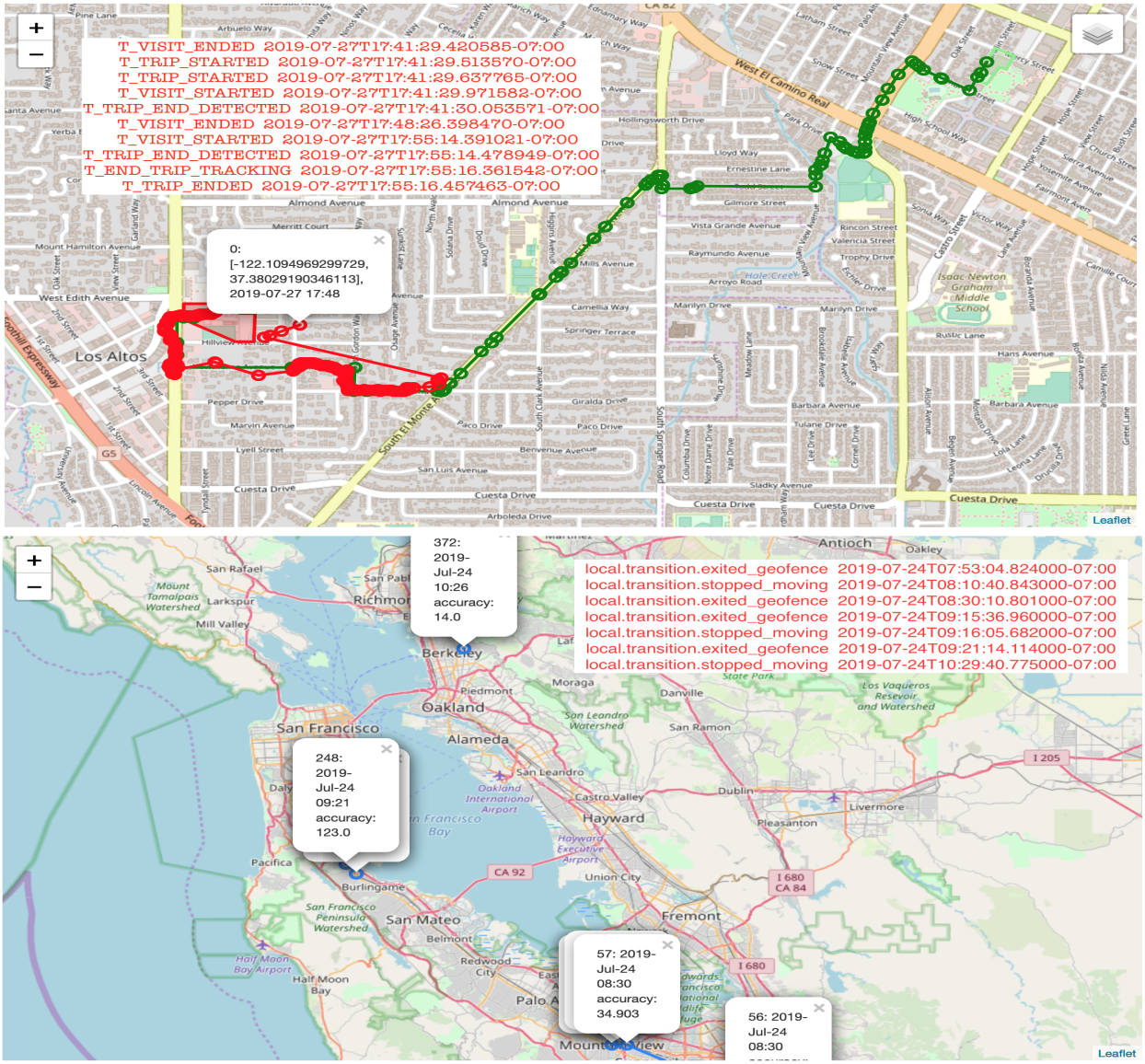

FIGURE 8 Examples of errors in segmentation captured by the segmentation metrics. Top – large error in the trip start time for an iOS `HAHFDC` run: The green line is the ground truth, red line is the sensed data. We received a visit end (trip start) transition at 17:41, but we detected a trip end within 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. Bottom – error in segmentation trip counts for an android `HAMFDC` run: On one multi-modal trip, we get multiple trip start and end transitions, largely corresponding to transit transfers. Note that at 8:30, there are two consecutive geofence exits (08:30 and 09:15) and an erroneous point in San Jose.

Mode inference

We detail through example the main causes of mode inference error on the NREL OpenPATH pipeline:

- The sensed points are collected at a delay (Fig. 8; Top). 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, Figures 5 and 6, the sensing delay at the start of a trip is classified under `NO_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 a section transition (Tab. 2) or trip transition (Fig. 8; Bottom). Again, this is captured via the temporal alignment process as FP and FN classification error.
- The sensed points are under reported (Fig. 9), 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 under-estimation of trip length, as discussed above, motivating the utility of using temporal alignment and trip measurement error in conjunction.

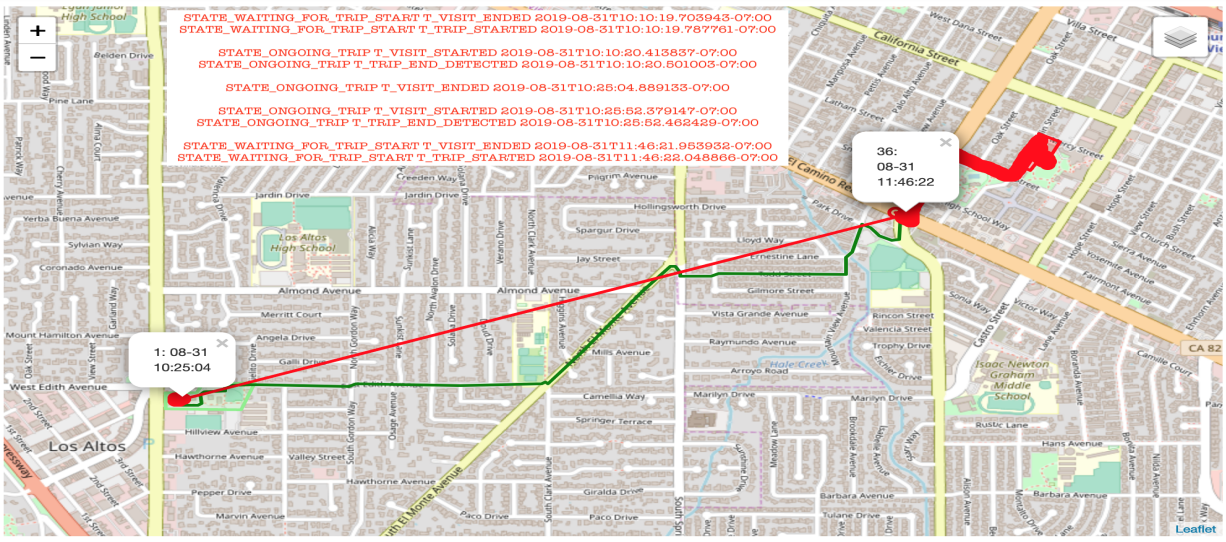

FIGURE 9 Example of bad data causing lingering missing sections. There are two ground truth trips, one from `10:07:27 -> 10:23:08` and another from `11:30:50 -> 11:52:38`. The sensed data includes an ephemeral trip from `10:10:19 -> 10:10:20` with no points and from `10:25:04 -> 10:25:52` with some points. Then most of the return trip is missed before getting a trip start at `11:46:21`. These trips are merged into one uni-modal section in the post-processing `10:25:07 -> 11:54:17`

	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 2 Example of how bad segmentation can lead to classification errors using an example fom 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`.

CONCLUSION AND FUTURE WORK

This paper presents a novel approach to evaluate multi-step 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 Prelipcean et al. (*9*), which is in turn based on time algebra alignment proposed by Allen (*10*). The use of temporal alignment lets continuous mode inference fit classic discrete multinomial classification paradigms, allowing for an measurement of *when in time* a mode prediction aligned with the ground truth. Additionally, our framework allows for seamless cross-sectional and longitudinal studies on arbitrary pipelines.

The modularity of our evaluation framework also supports expanding on current error models and distributions. Trip length measurements may take into account missingness factors as classified by Bähr et al. (*17*). Temporal alignments might be used to record distance, speed, or any arbitrary metric during overlap.

There are many avenues for future work to utilize or improve on our framework and subsequent evaluation results. 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 framework to study what factors 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 framework will allow for sensor-based multi-step pipelines outputs, in particular the interplay between trajectory segmentation and mode inference, to be understood in a continuous manner.

Acknowledgments

I would like to thank my research advisor and mentor, K. Shankari, for her invaluable guidance and support – helping me frame ideas, reviewing code, and answering any question (regardless of scope or substance) – while also serving as a reference for *any person* who aspires to work towards creating a greener, healthier world. I would also like to thank Michael Allen and Hannah Lu, whose valuable input was instrumental in completing this work.

To my parents: thank you for your unconditional encouragement and support in *all* of my ventures; without you none of this is possible.

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships Program (SULI) and the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy Vehicle Technologies, as part of the Core Tools program.

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.
2. Shankari, K., L. Boyce, E. Hintz, and A. Duvall, *The CanBikeCO Mini Pilot: Preliminary Results and Lessons Learned*. NREL, 2021.
3. 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.
4. McNally, M. G. and C. R. Rindt, The Activity-Based Approach. In *Handbook of Transport Modelling* (D. A. Hensher and K. J. Button, eds.), Emerald Group Publishing Limited, Vol. 1, 2007, pp. 55–73.
5. McNally, M. G., The Four-Step Model. In *Handbook of Transport Modelling* (D. A. Hensher and K. J. Button, eds.), Emerald Group Publishing Limited, Vol. 1, 2007, pp. 35–53.
6. 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.
7. SHANKARI, K., *e-mission: an open source, extensible platform for human mobility systems*. Ph.D. thesis, EECS Department, University of California, Berkeley, 2019.
8. Patterson, Z., K. Fitzsimmons, S. Jackson, and T. Mukai, Itinerum: The Open Smartphone Travel Survey Platform. *SoftwareX*, Vol. 10, 2019, p. 100230.
9. 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, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/13658816.2015.1137297.
10. Allen, J., Maintaining knowledge about temporal intervals. *Commun. ACM; (United States)*, Vol. 26, No. 11, 1983, pp. 832–843.
11. 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)*, 2020.
12. 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, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9671.2009.01152.x.
13. 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.), Springer, Berlin, Heidelberg, 2010, Lecture Notes in Computer Science, pp. 38–56.
14. Ranasinghe, C. and C. Kray, Location Information Quality: A Review. *Sensors*, Vol. 18, No. 11, 2018, p. 3999, number: 11 Publisher: Multidisciplinary Digital Publishing Institute.
15. 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.), Springer, Berlin, Heidelberg, 2011, Lecture Notes in Computer Science, pp. 350–367.

16. 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.
17. 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, _eprint: https://doi.org/10.1177/0894439320944118.
18. 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.
19. 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.
20. 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.
21. 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*, ACM, 2014, UbiComp '14, pp. 225–229.
22. 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.
23. 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.
24. Prelipcean, A. C., G. Gidófalvi, and Y. O. Susilo, Mobility Collector. *Journal of Location Based Services*, Vol. 8, No. 4, 2014, pp. 229–255.
25. Bishop, C. M., *Pattern Recognition and Machine Learning*, Springer, p. 38, 2006.
26. 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.
27. Shankari, K., M. Yin, S. Shanmugam, D. E. Culler, and R. H. Katz, *E-Mission: Automated transportation emission calculation using smart phones*. EECS Department, University of California, Berkeley, 2014.