MAKING SENSE OF THE NOISE: INTEGRATING MULTIPLE ANALYSES FOR STOP AND TRIP CLASSIFICATION

R. P. Spang[1]*, K. Pieper[1], B. Oesterle[1], M. Brauer[1], C. Haeger[2], S. Mümken[2], P. Gellert[2], J.-N. Voigt-Antons[3,4]

[1] Quality and Usability Lab, Berlin Institute of Technology, Berlin, Germany -
(spang, kerstin.pieper)@tu-berlin.de, (max.brauer, benjamin.oesterle)@campus.tu-berlin.de
[2] Institute of Medical Sociology and Rehabilitation Science, Charité - Universitätsmedizin Berlin, Berlin, Germany -
(christine.haeger, sandra.muemken, paul.gellert)@charite.de
[3] University of Applied Sciences Hamm-Lippstadt, Germany - jan-niklas.voigt-antons@hshl.de
[4] German Research Center for Artificial Intelligence (DFKI), Berlin, Germany

Commission IV, WG IV/4

KEY WORDS: GNSS, Analysis, Algorithm, Processing, Stop Trip Classification, Geometry.

ABSTRACT:

Mobility research is mainly concerned with understanding mobility on a higher level, including environmental factors, e.g., measuring the time out of home or tracking revisited places. This requires preprocessing the raw data obtained from GPS sensors, like clustering significant locations and distinguishing these from periods on the go. We introduce a new stop and trip detection algorithm to transform a list of position records into intervals of dwelling and transit. The system is based on geometrical analyses of the signal noise: Imperfect GPS data tends to scatter around an actual dwell position in a star-like pattern, and this imperfection is what we leverage for our classification. The system contains four independent classification methods, comparing different aspects of the geometrical properties of a given trajectory. If available, accelerometer readings can be used to improve the system's accuracy further. To evaluate the classifier's performance, we recorded a large dataset containing gold-standard labels and compared the classification results of our system with the results of Scikit Mobility and Moving Pandas. Our Stop Go Classifier outperforms the traditional distance/time-threshold-based systems. The described system is available as free software.

1. INTRODUCTION

Mobility researchers using GPS generally obtain raw coordinates and timestamps from their recorders. However, they are often far more interested in aggregated data, like significant locations, time spent out of home, or the number of revisited places. All of these rely on the ability to precisely distinguish stop from trip intervals and extract these from the raw coordinates and timestamps. Therefore, the conversion between these variables is fundamental in mobility research.

Spaccapietra and colleagues suggested the following concept for stops and trips: A temporal sequence of GPS coordinates where alternatively, the recorder position stays fixed and changes. A trajectory is a sequence of trips going from one stop to the next one (Spaccapietra et al., 2008). A stop hereby is defined as no movement larger than a distance d throughout a period t. The commonly adopted strategy to computationally identify stops and trips still involves this very combination of d and t; a distance and a time threshold to identify significant places (Ashbrook and Starner, 2002, Ye et al., 2009). Here, GPS records are grouped if they lie within a predefined radius and time.

Nurmi (Nurmi et al., 2009) defined multiple classes of algorithms for this task, differentiating radius-based, density-based, probabilistic clustering, and grid-based clustering algorithms. They further suggested a Dirichlet process clustering algorithm, based on Dirichlet process mixture models, a particular case of finite mixture models (Nurmi and Bhattacharya, 2008). Moving Pandas (Graser, 2019) and Scikit Mobility (Pappalardo et al., 2019) are probably the most widely adopted libraries for general mobility research and especially for mobility data preprocessing, and both provide modules for stop/trip detection.

When we planned the technical basis for a mobility intervention study, we tested several existing systems based on the time/distance-threshold approach. We chose simple and affordable smartphones as recording devices and tested existing solutions in the field. Using algorithms based on this principle, we observed, on the one hand, significant fragmentation of the identified stops due to the relatively large amount of signal noise. Fragmentation here is a single, actual stop detected as a set of multiple more minor, adjacent stops. On the other hand, we could only identify stops having a duration greater than a predefined time threshold, usually five minutes. Reducing this threshold lead to an increased number of falsely identified stops (false positives) and further fragmentation. Hence, the temporal resolution of this analysis was less than what we were looking for. This motivated us to develop a modern stop and trip classification algorithm.

Identifying stops in a GPS record is usually pretty easy for a human annotator: the GPS records scatter around the actual position when dwelling on a spot because of its imperfect signal. Records obtained from a trajectory through an environment are distinguishable - although the imperfect signal diverges from the actual position similarly (See Figure 1). This observation inspired us to create a new algorithm around the idea of investigating the signal patterns and, therefore, the geometric properties of the signal noise.

* Corresponding author

Figure 1. Example trajectory with stops and trips. Detected stops are highlighted in colour, and they show distinct shapes scattered around the actual dwelling position. Trips, in contrast, show clear paths between stops.

The goal of this manuscript is fourfold: Besides introducing a new classification system to the geo-FOSS community, we discuss the design decision of combining multiple independent analyses to form a majority-based decision on how to classify each GPS sample. This is a first in this context to the best of our knowledge. Further, we present ideas on how to geometrically analyze a GPS trajectory's shape by using the signal's noise and incorporating its properties into the classification decision. We demonstrate how each method is at least on par with other commonly used algorithms. Lastly, this manuscript provides benchmarks of the whole system using real-world GPS traces spanning an extended period to showcase the system's performance that we call the *Stop & Go Classifier*.

2. ARCHITECTURE

The *Stop & Go Classifier* is a Python class expecting a list of coordinates and corresponding timestamps to work on. Ideally, these coordinates are projected into a planar space. The classifier then scores each GPS sample as either a stop or trip, aggregates these labels to form stop intervals (consisting of a begin and an end time), and filters or merges outliers. The whole process is visualized in Figure 2.

Fundamentally, the algorithm is based on four different geometric analyses. We developed multiple approaches because any single analysis method was not robust enough in all of our test cases. We implemented them so they can overrule each other: a method with high confidence for its result can overrule an estimation of another method with lower confidence for a given case. Each analysis method is applied to a rolling window of subsequent GPS samples. This way, the shape of a small subset of the trajectory is evaluated independently. Each of the four analyses returns a score in the interval $[-1, 1]$, with -1 being most confident that the evaluated subset belongs to a trip and $+1$ being most confident that the subset belongs to a stop. Consequently, a score of 0 is inconclusive.

This score design allows for an easy combination through a simple average of all four scores. It allows overruling less confident scores from a few analysis methods through high confidence scores from others. Together, they form a classification decision for each GPS sample (Figure 2, section 3).

2.1 Four independent scoring methods

The following analysis approaches take a subset of a few consecutive samples to work on. Each method returns a score within the interval $[-1, 1]$ to express decision and confidence. Since each approach is different and internally arrives at different number spaces, each internal result is mapped to the target score interval to obtain comparable results. This mapping involves thresholds that we experimentally derived; more on this is explained in Section 3.2. The classification performance of each method is described in Section 3.4.

2.1.1 Width distance ratio This analysis compares two metrics of the given subset through a ratio. First, the cumulated euclidean distance between each subsequent pair of samples in the subset is computed. Second, out of all the points of the subset, the euclidean distance of the two points being furthest apart from each other is calculated (see Figure 3). The ratio of the path distance and the most exhaustive distance is the critical metric of this approach. In the last step, it is then translated to the target score interval $[-1, 1]$.

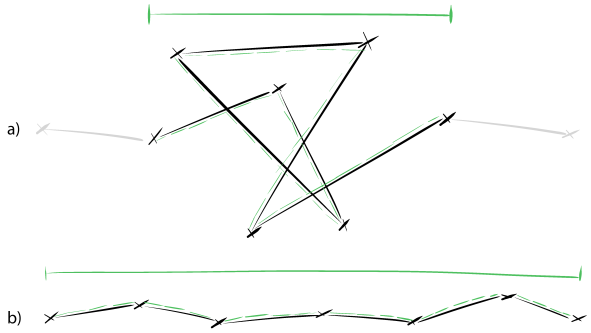

Figure 3. Sketch of the width-distance-ratio method: a) describes a stop; the solid green line is the distance between the two furthest points. This is compared against the path distance between all records of the stop (green, dashed line). b) shows the same comparison but for a trip segment.

Example: While on an ideal trip, e.g. walking along a straight line, the two values (path distance and longest distance between two points) are almost identical. Hence, the ratio between these is close to 1. However, when dwelling on one spot, the samples show a star-like pattern around that spot. The total path distance is approximately the number of samples times the average signal accuracy. For example, when the subset contains ten records and the accuracy is 15m on average, the path distance is 150m (although this is actually a stop, and the recording device did not move). Now, let us assume the distance between the two points being furthest apart is 30m (if the noise would cause two points to be precisely opposite each other). Then, the ratio between these values is $150m/30m = 5$. This value is much larger than the value we obtained from the trip example; hence this metric helps discriminate stop intervals from trip intervals.

2.1.2 Bearing This method splits the given subset into groups of consecutive triples. It spans a vector between the first and second point of any such triple and a second vector between the second and the third. It then computes the angle between the two resulting vectors, see Figure 4. This is repeated for all subsequent three samples. The critical metric of this approach is the average of all the angles. This way, it measures path continuity within the given subset.

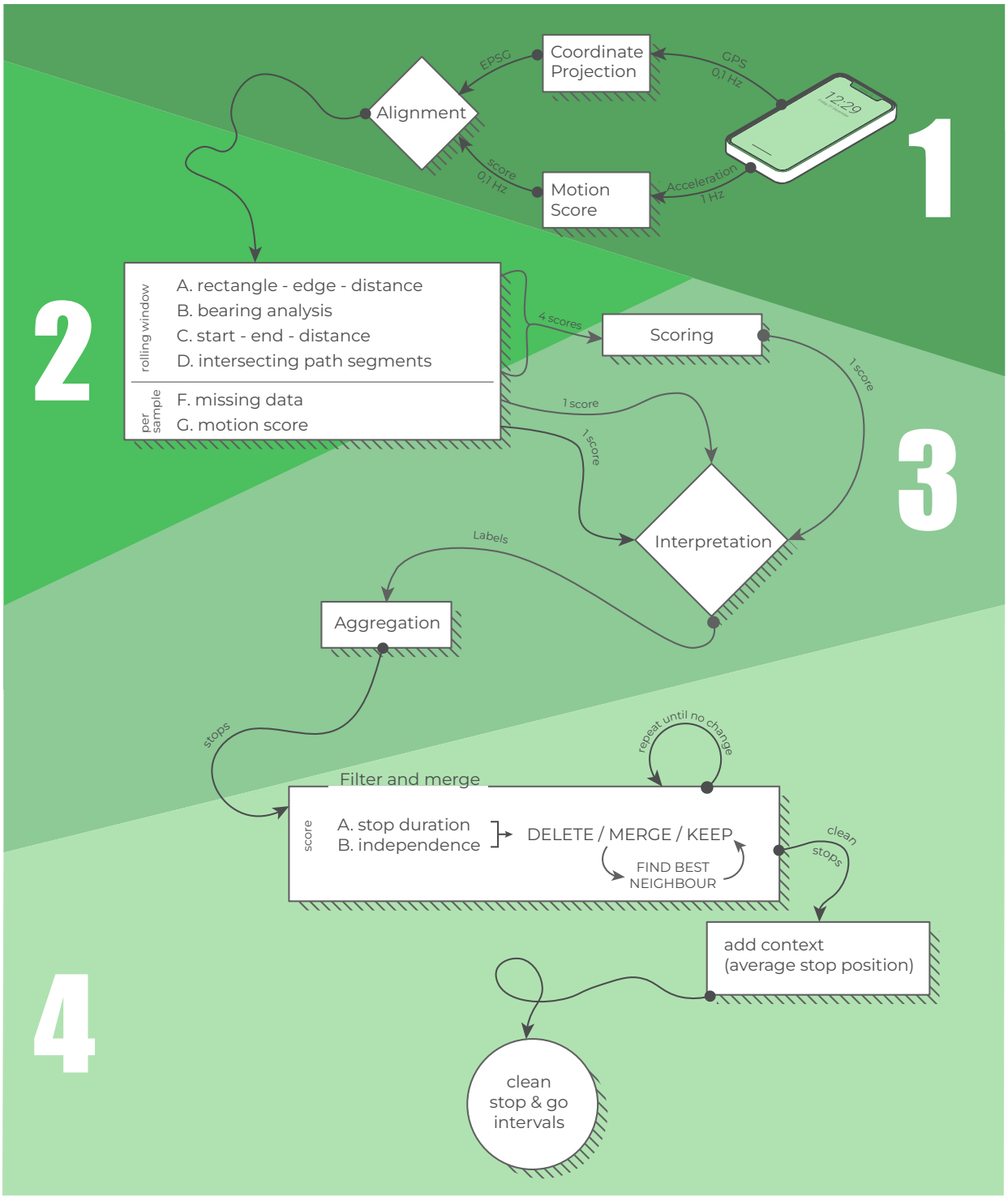

Figure 2. Flow-chart of the Stop & Go Classifier. The four stages represent (1) a simple preprocessing, (2) scoring a rolling window of subsequent samples using different geometric analyses, (3) classification of each GPS sample, and (4) filtering the obtained stop and trip intervals.

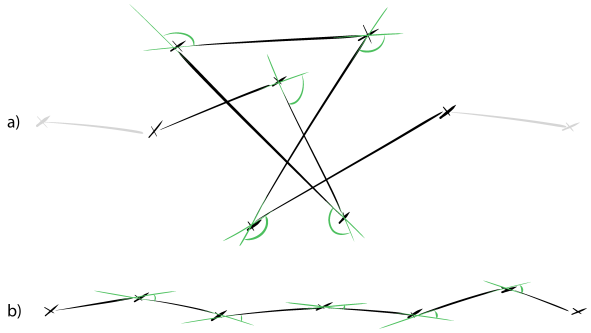

Figure 4. Sketch of bearing analysis method: a) describes a stop; changes in direction are measured as angles; drawn in green. b) shows the same procedure for a trip segment. The average angle of the subset is significantly larger for a stop than the average angle for a trip.

In an ideal trip, each point would follow the same direction as the previous points. Here, the average angle of such a set is close to zero. In comparison, a stop with its chaotic structure would have mostly large angles between its vectors. Thus, the average angle is quite large. Hence, this can be used to distinguish stops from trips.

2.1.3 Start-point end-point This method measures the distance between a given subset's first and last point. The position of the first and the last two samples is averaged to reduce the influence of outliers. Then, the distance between these two averages is computed. Within an ideal stop, this distance is close to zero (see Figure 5). However, when applied to a strait trip interval, this distance is almost as long as the cumulative euclidean distance of the trip itself.

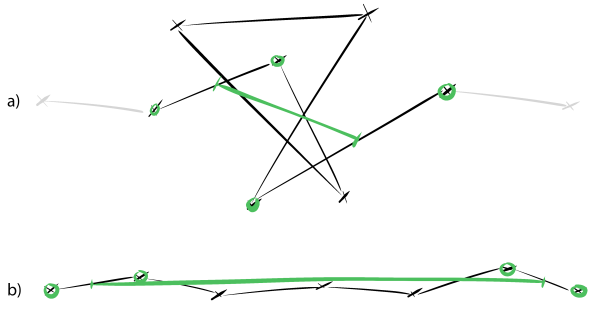

Figure 5. Sketch of the start-point end-point method: a) describes a stop; the solid, green line is the distance between the stop's first two and last two points. b) shows the same procedure for a trip segment; the distance between start and end is significantly shorter for a stop than for a trip.

Note that this approach is structurally similar to the traditionally used approach introduced by (Ashbrook and Starner, 2002). The critical variable time t is given through the window size (the number of samples over a specific interval). The distance between start and end is then comparable to the diameter of the classic algorithm.

2.1.4 Path segment intersections This last method counts the intersections between all path segments within the given subset. A path segment is a line between two subsequent position samples. It is unlikely to have multiple segments intersecting each other within a trip. However, samples of a stop scatter around the actual position because of the signal noise. This jitter of the recorded samples causes frequent intersections (see Figure 6).

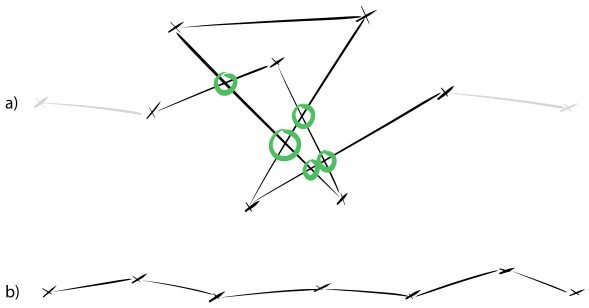

Figure 6. Sketch of the path segment intersections method: a) describes a stop; green circles highlight intersecting path segments. b) shows a trip segment. Here, no intersection occurs.

It is possible to have intersecting elements within large windows of trip data. For example, crossing the same street multiple times would cause intersections too. Therefore, using a relatively small window size is essential for this method.

2.2 Device motion

Apart from coordinates and timestamps, the system can process a list of motion score values. These quantify the recording device's physical movement at any given time. While we suggest how to compute and incorporate such measures into the analysis, interpreting such a measure can be reconfigured. During sample acquisition, recording devices such as smartphones can also easily record three-dimensional accelerometer data. For our demonstration and benchmark, we acquired accelerometer readings at 1Hz. In a preprocessing step, we computed a motion score (MS) that expresses the amount of physical movement. We propose Formula 1 to transform 3D accelerometer readings of a given subset into a single scalar. Applying this as a rolling window to all acceleration data allows quantifying the device's motion at any given time. This can then be used to distinguish intervals of the device being in a fixed position (e.g. on a table or in a jacket) from the device being moved (e.g. in a pocket while walking or in a driving car).

$$\text{motion\_score}_i = \sum_{dim}^{\{x,y,z\}} \sigma(dim_{i-t} \ldots dim_{i+t}) \quad \text{with } t = 60s \tag{1}$$

Formula 1 describes the simple sum of the three standard deviations of the accelerometer values on all three axes. In this example, we consider a two-minute interval around the point in time of interest. For example, the motion score at 10 am on a given day considers all accelerometer readings between 9.59 am and 10.01 am. It computes each axis's standard deviation σ and adds all three values together. This procedure is agnostic of the device's position, whether lying flat on a table or upside down in a backpack. If the device is not moved, the σ of all

three axes will be close to zero (only capturing sensor noise). However, the motion score would be significantly larger if it was constantly moved on any or multiple of the axis.

We use this motion score to determine periods of no movement for our stop and trip detection. Fundamentally, if we know that the device (and hence its environment) was not moved, we assume a stop. Because any significant movement will involve a motion to the device, this analysis significantly reduces false trips and fragmentation. However, the *Stop & Go Classifier* does not rely on the availability of a motion score and falls back on the described methods if none is available.

2.3 Missing data

The last classification support of the system is concerned with interpreting missing data. Since the GPS signal is easily lost underground or in buildings, gaps in the data are a common issue (especially when the sample accuracy is capped at recording time). For example, when analyzing the trajectory of an individual working at a large office building, one might follow their path sample by sample from home to work, then observe a nine-hour gap in the data, and finally a similar but reversing path from the office back home. As our described approach only inspects the geometrical properties of the samples, we would miss a significant stop. In this example, the geometrical shape of the samples would look like going back and forth, without any distinct pattern to recognize. Because of this, we include a missing data analysis. This considers two aspects: the time between two subsequent samples and their spatial distance. The latter is essential not to mistake a travel period for a stop. The ratio of these two dimensions equals their velocity (distance/time). Hence, we inspect the velocity between all consecutive sample pairs. For the described example of a prolonged stay in a building without records, the velocity between the last sample recorded before entering the building and the first after leaving it would be close to zero. In a different scenario, e.g. a flight, we would also observe gaps in the recorded data. However, the distance per time unit would be significantly more prominent, hence the velocity between the two samples around the data gap. Thus evaluating the velocity between the samples helps interpret missing data periods.

2.4 Aggregation, filter & merge

With all these described scores in place, the system classifies each sample as either stop or trip. Subsequently, this sample-by-sample information is transformed into a list of stops by detecting changes between labels. As a result, a list of stop intervals is obtained. Attributes include begin and end timestamps and a stop position. Each position is the average of all sample positions within the interval between begin and end time.

The system's last step is cleaning this aggregated result list up. Therefore, each stop interval is inspected regarding duration and is compared against its neighbours regarding the temporal and spatial distance between them. There are three options to choose from: Either a stop can be kept as it is, it can be merged with another stop-interval nearby, or it can be discarded as irrelevant (Figure 2, section 4). This analysis reduces fragmentation and caters to a cohesive, clean result set.

3. BENCHMARKING

To test the accuracy of our analysis approach, we benchmarked the system against the built-in methods for stop and trip detection of Moving Pandas (Graser, 2019) and Scikit Mobility (Pappalardo et al., 2019). We investigate sample-by-sample classification metrics (accuracy, precision, recall/sensitivity, specificity, and F1) and stop/trip interval-specific metrics (number of stops and trips, number of missed stops and trips, and quantify fragmentation).

3.1 Dataset

To evaluate the performance of our classification system, we needed an accurately labelled dataset containing real GPS traces and annotations about the stops and trips of the person recording the data. The STAGA-Dataset was created to evaluate the classification performance of stop/trip detection algorithms (Spang et al., 2022). This dataset provides smartphone recorded GPS data (0.1Hz), including accelerometer readings (1Hz). The recorded trajectories span over 126 days of everyday life. The dataset contains 122,808 GPS samples (78,900 labelled as stops and 43,908 labelled as trips). A detailed movement diary contains 692 annotated stops.

This movement diary acts as the ground truth for comparing different classification algorithms. Using the provided accelerometer data of the recording device, we computed a motion score as described in Section 2.2 to investigate its performance implications. However, since our two reference libraries do not include options to interpret acceleration data for stop/trip detection, we report results with and without using the motion score.

3.2 Parameters

As in most optimization problems, there is a wide variety of parameters to optimize for. Popular metrics like the F1 score, specificity, and sensitivity are sample-level metrics. In our context, this means testing if a GPS record is correctly labelled as a stop or a trip. However, other priorities might be considered depending on the use case. For example, fragmentation becomes a significant problem quickly when investigating the number of stops one has made in a given time. Since many classifiers struggle with fragmentation, the raw number of stops is often overestimated. Hence, sample-based metrics will not capture all important aspects when tuning a classifier. Instead, optimizing for a minimum fragmentation or as close to a perfect number of stops could be more desirable.

Since most systems have multiple settings to adjust for a given dataset, we employed a grid-search approach to determine the best classification performance parameters. For a fair comparison against the other systems, we employed the same procedure to optimize the parameters of Scikit Mobility (version 1.2.3) and Moving Pandas (version 0.9rc3). Scikit Mobility yielded best results with 2.175 *minutes for a stop* and .048 *spatial radius km*; Moving Pandas with a *min duration* of 175 sec and a *max diameter* of 104 m. Similarly, we tuned the parameters of the *Stop & Go Classifier* to yield optimal results.

When tuning the parameters of the three systems, we focused on improving the balanced accuracy score. This captures the average between the true positive rate (ratio between stops that were correctly classified as a stop and all classified stops) and the true negative rate (ratio between trips that were correctly classified as a trip and all classified trips).

3.3 Results

Table 1 summarizes all results. For a fair comparison, we report the classification performance of the *Stop & Go Classifier*

without considering the motion score since our two reference systems do not incorporate accelerometer data. However, to showcase its influence, we report the performance of our classifier with and without the motion score.

	Scikit Mobility	Moving Pandas	Stop & Go Cl. w/o Motion Score	Stop & Go Cl. w/ Motion Score
Correct	93.4%	91.65%	96.37%	96.84%
Bal. accuracy	.931	.922	.966	.968
F1 score	.948	.933	.971	.975
Correct stops	94.06%	90.25%	95.76%	96.96%
True stops	74,208	71,204	75,550	76,495
False stops	3,421	2,558	1,112	1,477
Correct trips	92.21%	94.17%	97.47%	96.64%
True trips	40,486	41,349	42,795	42,430
False trips	4,686	7,690	3,347	2,399
Stop count	1,071	1,575	753	670
Missed stops	27	81	17	17
Fragmented s.	145	255	67	25
Trip count	1071	1575	753	670
Missed trips	27	17	28	33
Runtime	2.71s	47.17s	36.79s	26.69s

Table 1. Sample- and cluster-based classification performances of the two reference systems and the *Stop & Go Classifier*. Best performances per metric are highlighted in green. Balanced accuracy is abbreviated as *Bal. accuracy*. *Fragmented s.* refers to the number of stops including multiple fragments instead of one cohesive stop.

3.4 Individual algorithms

Although the *Stop & Go Classifier* consists of several different analysis methods, we measured the performance of each method independently. Table 2 lists the balanced accuracy and F1 score for each of the four scoring algorithms independently. It also shows the performance change when all algorithms' results are combined and showcases the influence of the Motion Score and the missing data analysis.

Method	Balanced Accuracy	F1 Score
Width distance ratio	.943	.947
Bearing analysis	.938	.948
Start-end-distance	.941	.943
Path segment intersections	.941	.943
All 4	.96	.963
All 4 + motion score	.965	.97
All 4 + missing data	.966	.971
All 4 + missing + motion	.968	.975

Table 2. Sample-based performance analysis of the individual scoring methods and their combinations. The combination of the four analysis methods performs better than the four individual methods because the strategies can compensate for each other in cases of uncertainty.

4. DISCUSSION

As discussed in Section 3.2, there are many different metrics to quantify classification performance. We distinguish two main categories: sample-based and cluster-based. Sample-based metrics compare the classification of each GPS sample to the ground truth label provided by the dataset. These metrics are listed in the first part of Table 1. As indicated by the highlight, the *Stop & Go Classifier* outperforms the reference implementations in all these metrics.

The improvement over the second-best system ranges from 1.7% (true stops) to 3.5% (balanced accuracy). However, the absolute number of falsely classified stop samples is reduced remarkably (56,5%) without compromising the number of false trips (28,6% better). So judged by traditional sample-by-sample metrics, the *Stop & Go Classifier* provides the best classification performance of the three systems.

The second category of analysis metrics is cluster-based, i.e. it compares the resulting stop and trip clusters (a set of subsequent samples belonging to the same stop or trip) against the intervals of the ground-truth diary. This allows quantifying how well the *duration* of a stop or trip is represented in the classifiers' outputs. Additionally, we investigated the frequency of stop fragmentation, where one stop in the dataset is recognized as multiple, more minor stops. This is an important aspect to consider when evaluating classifiers: even strong fragmentation might not influence the sample-by-sample based performance metrics drastically, but it obviously would affect, e.g. the number of important places, as fragmentation would increase the number of stops potentially drastically. In our comparison, the *Stop & Go Classifier* comes closest to the actual number of stops (753/692) and trips (753/691) *and* has the least number of fragmented stops. While a good result was to be expected, given the good sample-by-sample results, it is worth noting that the substantially reduced amount of fragmented stops (about one-third of the second-best system) is most probably earned through our sophisticated merging mechanism. This component could also be applied to other classifier systems. However, our implementation missed more trips than Moving Pandas' stop detection algorithm. Further analyses are needed to understand the exact mechanics causing this difference. Lastly, it is worth mentioning that Scikit Mobility's implementation outperforms the other systems by order of magnitude regarding runtime.

Our results show that integrating multiple analyses helps strengthen the classification and make it robust against scenarios in which a single method might fail. Combining the four metrics further improves the balanced accuracy of about 1.7% over the best of the four individual results (see Table 2). Each of the four individual methods is based on geometrical analyses that are all on par or even more robust than the second-best reference system (regarding balanced accuracy). This makes a strong case for geometrical trajectory analyses in the given context.

The comparison is based on a collection of metrics we deem most interesting for choosing algorithms to detect stops and trips from raw GPS data. While this benchmark is not complete, it should motivate considering the *Stop & Go Classifier* for stop and trip identification tasks. It might be worth investigating additional ones for particular use-cases. Other metrics might conclude that different software is relevant for a given task.

4.1 Limitations

We tested our system against two popular and readily available Python libraries, although the academic literature knows other algorithms for similar purposes. For example, (Joo et al., 2020)

lists several segmentation methods available for the R language. Further comparisons against different algorithms should be carried out to get a more meaningful impression of how well our system performs.

Furthermore, while we tuned all systems to perform best on our dataset, more benchmarks are needed, including different datasets recorded under different conditions and a variety of sample parameters (e.g. much higher or lower sample frequencies and sensor accuracies).

Lastly, the comparison against Scikit Mobility highlights how time-efficient stop/trip classification can be performed. While our code is production-ready in principle, it might be worthwhile to redesign core components focusing on performance and keeping the same level of expressiveness.

4.2 Future work

While the dataset we used to benchmark the systems is large enough to provide evidence of effectiveness, it only contains data from one specific recording device (captured at 0.1Hz GPS sample frequency with a maximum accuracy of 25m). More diversity would be desirable to showcase the algorithm's generalizability. During the system's development, we tested various synthetic datasets but focused on the real-world data for this benchmark. However, using synthetic data and systematically investigating different trace parameters (like frequency, accuracy, or gaps) would help identify current weaknesses.

4.3 Conclusions

We demonstrate a new algorithmic approach to transforming raw spatio-temporal coordinates into intervals of stops and trips. Two aspects are most relevant: we employ four different algorithm approaches to independently score each sample as either stop or trip. These methods' results are then combined to form a single result. This allows to compensate for unconfident estimations and strengthens the overall classification performance. Second, all these algorithms work on geometrical analyses of the shape of the trajectories. Since the position data suffers from signal noise, we base our classification around the geometrical analyses of the properties of the signal. This requires little to no preprocessing and renders the need for signal filtering or smoothing obsolete. To evaluate the classification performance, we classified stops and trips in a dataset containing trajectories of over four months, including annotations of over 650 stops. Our *Stop & Go Classifier* is at least on par with traditional methods and outperforms the two comparison systems in most aspects.

5. CODE AVAILABILITY & RESULT REPRODUCIBILITY

The *Stop & Go Classifier* is free software under a BSD 3-Clause license. The repository includes a reference implementation of the algorithm and small usage examples[1]. We provide a second repository containing all scripts used to generate the results of this manuscript[2]. The software of this second repository was also used to estimate optimal parameters for all three systems to yield the best performances on the given dataset.

[1] github.com/rgreinacher/stop-go-classifier
[2] github.com/njamster/stop-trip-evaluation-framework

6. ACKNOWLEDGEMENTS

We thank Bartholomeus Tümmler for his background advice on coordinate reference systems. We thank Purna Dutta for her help preparing the STAGA dataset to be applied in this benchmark. We further thank Sonia Sobol for her artwork visualizing the flow chart of the algorithm.

REFERENCES

Ashbrook, D., Starner, T., 2002. Learning significant locations and predicting user movement with gps. *Proceedings. Sixth International Symposium on Wearable Computers,*, IEEE, 101–108.

Graser, A., 2019. MovingPandas: efficient structures for movement data in Python. *GIForum*, 1, 54–68.

Joo, R., Boone, M. E., Clay, T. A., Patrick, S. C., Clusella-Trullas, S., Basille, M., 2020. Navigating through the r packages for movement. *Journal of Animal Ecology*, 89(1), 248–267.

Nurmi, P., Bhattacharya, S., 2008. Identifying meaningful places: The non-parametric way. *International Conference on Pervasive Computing*, Springer, 111–127.

Nurmi, P. et al., 2009. Identifying meaningful places.

Pappalardo, L., Simini, F., Barlacchi, G., Pellungrini, R., 2019. scikit-mobility: A Python library for the analysis, generation and risk assessment of mobility data. *arXiv preprint arXiv:1907.07062*.

Spaccapietra, S., Parent, C., Damiani, M. L., de Macedo, J. A., Porto, F., Vangenot, C., 2008. A conceptual view on trajectories. *Data & knowledge engineering*, 65(1), 126–146.

Spang, R. P., Pieper, K., Oesterle, B., Brauer, M., Haeger, C., Mümken, S., Gellert, P., Voigt-Antons, J.-N., 2022. The STAGA-Dataset: Stop and Trip Annotated GPS and Accelerometer Data of Everyday Life. *Proceedings of FOSS4G, Florence, Italy*.

Ye, Y., Zheng, Y., Chen, Y., Feng, J., Xie, X., 2009. Mining individual life pattern based on location history. *2009 tenth international conference on mobile data management: Systems, services and middleware*, IEEE, 1–10.