Region Formation for Efficient Offline Location Prediction

Movement prediction offers valuable insight into when it's beneficial for devices to interrupt users. A new offline location prediction approach for low-power devices represents user mobility patterns as an optimal set of geographical regions, increasing both precision and recall.

Ian Craig and Mark Whitty
University of New South Wales

Each of us follows regular daily and weekly routines, making 93 percent of our movements theoretically predictable.[1] Although surrounded by countless shops and attractions, we visit relatively few; the paths we follow between these destinations are thus predictable. Whether we're moving around a city, campus, or even our home, context-aware services that can predict our movements can provide us with useful insights regarding subsequent actions. For example, using such a service, our devices could cache information[2] and adapt user interfaces to show us what we need before we need it.[3] Or, when we deviate from expected routines, our digital assistants could use the insights to provide timely reminders.

As wearables and embedded devices become common, we have more opportunities to use services that exploit predictions. However, today's approaches to location prediction have several limiting factors. In particular, there are three key problems for location prediction using low-power devices:

- complexity and power consumption,
- privacy, and
- scope and data sparsity.

Techniques to mitigate each of these problems exist, but none are without limitations.

We propose a fundamental change in location prediction. Rather than constantly uploading the user's location, our approach enables offline prediction. Devices would request information only about destinations and could even cache this information, greatly improving privacy and reducing power consumption through network usage. Here, we present our novel region formation algorithm and evaluate the performance gains from using regions in existing prediction algorithms.

Problem Overview

When location and movement prediction is successful, it offers many opportunities to assist users. However, several challenges must be addressed to make this prediction practical for today's low-power mobile devices.

The first challenge is to reduce algorithm complexity and power consumption. Embedded devices have extremely limited resources, making many prediction algorithms unfeasible. Further, current battery technology is the biggest constraint on mobile applications.[4,5] Location prediction in the cloud is equally problematic, because a constant connection can use even more energy than processing offline, especially on poor connections.[6] Even offline, frequently recording the user location prevents a device from switching off the core clock and certain peripherals. The resulting

Published by the IEEE CS ■ 1536-1268/17/$33.00 © 2017 IEEE

impact on the battery is far more than just power used to record and store the data—frequently recording and storing sensor data reduces power efficiency by 10 times when compared to when the device is idle.[5]

The second challenge is user privacy. Any system that regularly uploads a user's location poses serious privacy concerns. Currently, no consistent approach exists in pervasive computing to ensure data security.[4]

Finally, we must contend with the challenge of scope and data sparsity. Prediction algorithms struggle to understand new paths, and previous trips cover few of the possible paths between destinations.[7] Relying on information sources such as street maps limits both the scope of prediction and the possible applications. This information is also costly to store and associate with live recordings.[8]

Techniques to mitigate each of these problems exist, but none are without limitations. For example, Baik Hoh and his colleagues[9] demonstrated that user's trips can be anonymized in large datasets by deleting small sections where they overlap with other users; trips, but this can only occur once data reaches the cloud, and it would still render data useless for personalized predictions. Likewise, unknown areas can be dynamically mapped to simplified graphs using techniques such as Support Points,[10] but this doesn't solve the computational overhead of associating locations with this graph, and there are still many cases where data would not be able to be associated with the graph.[11]

Solution Overview

Our approach provides a more natural, comprehensive solution to these key problems in that it

- has a low requirement for location accuracy,
- automatically regulates frequency of location recordings,
- has low computational complexity and memory usage,
- works offline,
- requires no context or map, and
- naturally generalizes new paths.

Our approach achieves this by dividing an area into the fewest possible geographical regions, while still fully representing the different paths a user takes. Region boundaries are formed at locations where users are more likely to change their potential destination, such as where paths diverge.

We consider regions to represents areas in which predictions will provide valuable insight. No valuable information is gained from knowing exactly where in a region the user is, and predictions occur only at region boundaries. This allows for a reduction in location accuracy and frequency, letting devices sleep and consume far less power.

Regions also significantly reduce the state space of the prediction problem. Areas are represented using a grid, similar to existing approaches.[11–13] However, while our grid typically has 15,000 cells, our average user has only 71 regions. This reduction enables offline prediction and increases training data per state.

Previous techniques for producing simplified street maps[10] achieve a similar reduction in complexity, but without the benefits of generalizing nearby streets and reducing location accuracy and frequency. In fact, the generalization that regions provide make them extremely useful—even for higher-powered devices—in scenarios where the area is unmapped or location accuracy is low, such as walking around your home.

Because our lives often revolve around weekly routines, the region model would require only weekly updates. Although model generation is computationally expensive, the infrequent task could easily be done in the cloud, providing a much more optimal utilization of resources. Devices could upload location history and thereby update the offline prediction model at a convenient time, such as at home while charging and on a secure network.

TABLE 1
Comparison of destination detection algorithms for different stay durations.*

	Greedy		Hierarchical	
	10 min.	30 min.	10 min.	30 min.
Distance (m)	115	88	92	86
False positives**	41	25	16	9

* In all metrics, lower is better, but should be considered only as a relative comparison.
** False positives are locations detected as destinations that don't match any of the 102 ground truth destinations.

Data Sources

We define a *record* as a single location and time measurement (that is, latitude, longitude, and time/date). Data sources are interpreted only as a stream of records. We used four data sources: GeoLife GPS Trajectories,[14–16] Microsoft's MSR GPS Privacy Dataset 2009,[17] Google's Location History service, and a high-resolution GPS log from a university campus.

We define a *stay* as an event in which the user is relatively stationary for a significant period, and a *destination* as a location the user frequently stays in. We aim to predict the user's trips between stays. Stays were identified using Natalia Marmasse and Chris Schmandt's approach to addressing GPS dropouts,[18] and Niels Brouwers and Matthias Woehrle's approach to dwelling.[19] Destinations are then found by clustering stay locations. We compared a greedy algorithm described by Daniel Ashbrook and Thad Starner[2] with a hierarchical approach by Ramaswamy Hariharan and Kentaro Toyama.[20]

We asked five users to identify all frequent destinations, producing a ground truth set of 102 destinations across multiple cities. Table 1 shows the mean distance between each detected destination and the corresponding ground truth, along with the number of false positives. The hierarchical approach

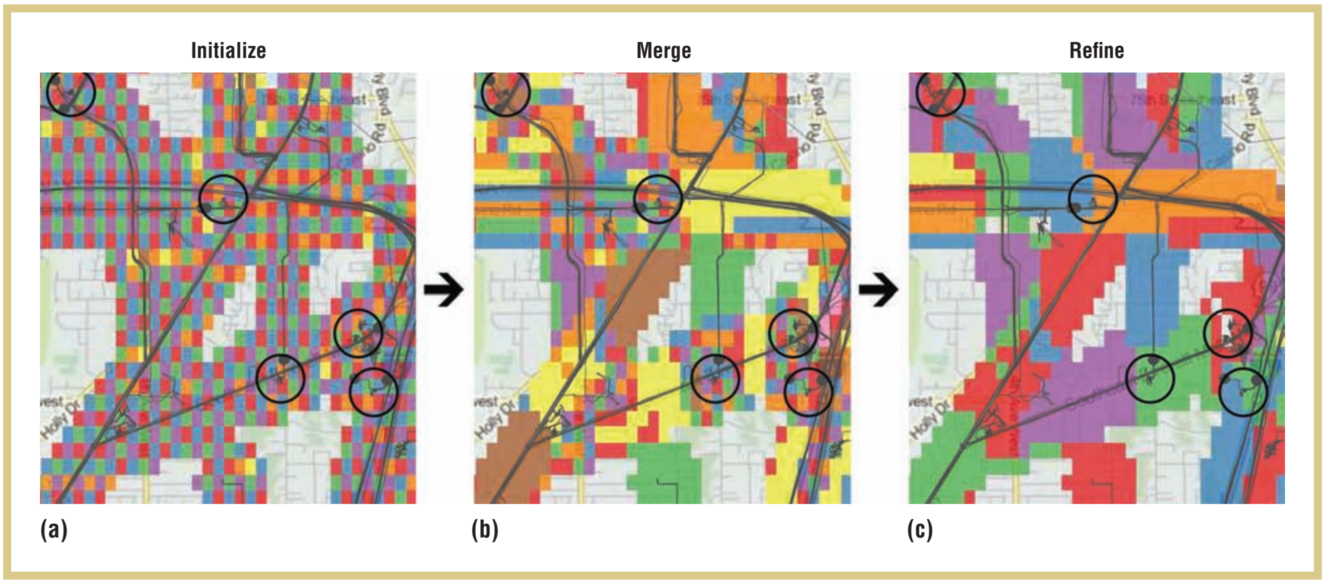


Figure 1. The approach's three stages. These graphics show regions covering part of a user's area after each of the three stages of region formation: (a) initialize, (b) merge, and (c) refine. The black circles represent destinations, and gray lines depict trips. Regions are colored areas, and their colors are chosen only so that no adjacent regions have the same color. The merging phase forms large regions along highways, as the initial regions here are passed by the same trips. Between these, small regions remain until the refinement stage, during which they are merged to simplify the region map.

produced marginally more accurate locations and significantly fewer false positives.

For the following results at city scale, a stay is defined as 10 minutes. Destinations have a radius of 200 meters and were visited at least five times. These values were chosen empirically by examining destinations for 20 users. Various applications would have different requirements. As stay durations, visit minimums, and destination sizes are increased, the problem becomes easier, resulting in lower variance and better predictions. Our approach excels in the more challenging short and frequent trips that the chosen values define, arguably providing the most useful context for location services.

Region Formation Algorithm

Our area of interest is a bounding rectangle encompassing all trips the user makes between destinations. Our algorithm divides this into regions such that all areas within a region have a similar proportion of previous trips ending at each destination. Regions are simply a representation of area and can't overlap.

As Figure 1 shows and we describe below, there are three distinct stages. Starting with a dense grid of regions, a merging process builds larger, more general regions without discarding important information. Larger regions greatly reduce the state space for prediction and accumulate more training data, improving performance by reducing the influence of any single piece of evidence.[21]

Initialization

In the first stage (Figure 1a), the area is quantized into a grid of square cells, which are used to describe the shape of regions. As Figure 2 shows, as grid resolution increases, the number of regions formed drops sharply; it then stabilizes as it becomes possible to form the desired region shapes. This drop occurs at a higher resolution for users with smaller areas of interest; we thus set the bounding rectangle's shorter side to 100 cells, placing city users in Figure 2 at 750–1,500 cells/degree latitude. This choice also makes the algorithm scale independent, performing equally well for areas the size of a campus or city.

Initialization begins by creating individual regions to cover each grid cell in the target area. For each region, trips passing nearby are inspected and the distance from the region's center c to the nearest record r is calculated. A radial basis function φ determines the trip's membership value μ_t for the region:

$$\mu_t = \varphi\left(\left\|c - r_{nearest}\right\|\right).$$

Trip memberships are used to identify and merge similar regions in the merging and refinement stages. The radial basis function smoothes trips to accommodate uncertainty in localization. A Gaussian function centers the regions along the most frequent path more appropriately than a triangular or trapezoidal function. We chose a variance of 1.5 grid cells to produce large regions without incorrectly consuming smaller adjacent regions, optimizing precision and recall of final predicted output.

Merging

For each region, a discrete probability distribution $P(X)$, representing the probability that a trip t through this region will end at each destination d, is calculated as the normalized sum of trip memberships ending at each destination:

$$P(X) = \frac{\Sigma_t\{\mu_t | t \rightarrow d\}}{\Sigma_t \mu_t}.$$

We select one region at a time, and, for each of its adjacent regions, we attempt a merge. The absolute error ε_{ab} between the two region's distributions is calculated, and a merge is performed if the error is below threshold ε_{max}:

$$\varepsilon_{ab} = \Sigma_d \left| P(X_a = d) - P(X_b = d) \right|.$$

Merging takes the maximum membership μ_t for each trip from the two regions, and the new region covers the area occupied by both. $P(X)$ is recalculated using the new μ_t values as above. The process is repeated until a complete pass is made with no merges. To select ε_{max}, precision and recall of final predictions was evaluated (see Figure 3) and 0.13 was selected to minimize the number of regions while maximizing precision and recall.

We tested three options for the order in which regions were selected along with a *Control*, which performed no merges, and a more advanced model, *Random with origin*, which based $P(X)$ on both the origin and destination of the trip. Table 2 shows all of the results. Random ordering proved a minor advantage, and we used it in all of the following tests.

We investigated the effect of including the time spent or number of records in each region, but in both cases this resulted in too little training data for small regions.

Refinement

When merging is completed, small regions between and around the significant regions remain, as Figure 1c shows. Insignificant regions—as

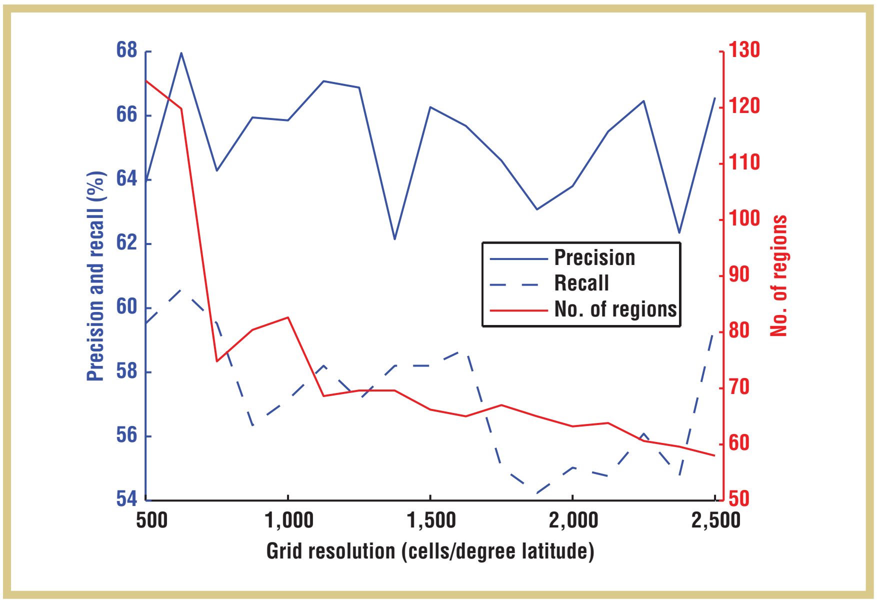


Figure 2. Precision, recall, and number of regions as grid resolution increases. This data is drawn from tests of 150 users at a city scale.

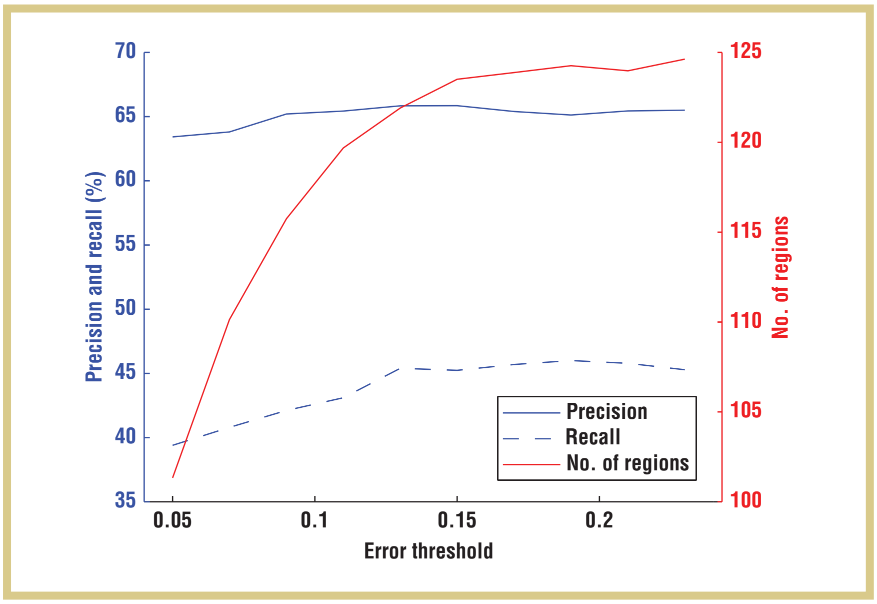


Figure 3. Average precision, recall, and number of regions as the total error threshold is varied for all users. Here, we chose 0.13 as the error threshold.

TABLE 2
Results of merge order tests.*

Order	No. of regions	Precision (%)	Recall (%)
Sequential	113.2	62.06	44.45
Random	117.6	62.59	45.00
Best-first	126.4	61.45	44.10
Control	1530.1	53.89	66.02
Random with origin	59.8	61.33	34.52

* Performed using all users in GeoLife, MSR Privacy, and Google Location History sources with more than 150 trips.

defined here—are either merged into adjacent significant regions or deleted if there are none to merge with:

> Any region covering only 3 cells or fewer
>
> OR
>
> Any region covering less than 10 cells AND
> a member of less than 10 trips
> OR
> containing no membership value of 0.5 or greater

This process forces generalization, cleaning up regions that cause overfitting and reducing the number of regions. This is vital to enabling prediction on low-power devices, because it further reduces the state space for predictions.

With this definition, we observed a 4 percent increase in precision and more than tenfold reduction in the number of regions. For both rules, we saw an initial steep drop in region number and a rise in precision that flattened off quickly around the chosen values. More aggressive rules had little impact and eventually reduced precision.

Location Prediction

We have shown how our novel region formation algorithm breaks the user's area into regions. To measure the improvement these regions have on existing location prediction models, we tested several region- and nonregion-based models and compared them statistically.

For each location record, the model is given an input (described below) and produces an estimate—that is, a probability distribution for the likelihood of the trip ending at each destination.

Model configurations define which of the following inputs are included:

To evaluate how regions affect predictions, we defined several model configurations both with and without regions.

- *Origin*: destination where this trip started.
- *TwoOrigin*: current and previous origin.
- *Region*: region the user is currently in.
- *TwoRegion*: current and previous region.
- *Time*: current recurring time interval.

Recurring time intervals split the week into 15-minute sections. As the interval is shortened, precision and recall increase. We selected 15 minutes empirically as a compromise between improved predictions and increased complexity.

The model is composed of states, defined by some combination of input values. Each state is associated with a distribution, which is trained by counting the trips to each destination matching this state. If the current input exactly matches a state, this distribution is the estimate. When an input is received but is incomplete or doesn't exactly match a known state, the model aggregates all partially matching states by summing the number of trips to each destination before normalizing. This prevents less-certain states from skewing the estimate.

This prediction model is intended only to demonstrate relative performance gains from using regions and would not compete with more complex algorithms. The results we can achieve using regions with such a simple model also demonstrate how regions enable accurate location prediction on low-power devices.

Model Configurations

To evaluate how regions affect predictions, we defined several model configurations both with and without regions, named according to which of the above inputs they include. Each is explained here, and the results of testing all of these configurations is shown in the following section.

Nonregion model configurations. We begin with a benchmark of five configurations representing existing approaches. *Origin* and *TwoOrigin* mimic the basic first- and second-order Markov model as used by Ashbrook and Starner.[2] These two models update their estimates only when a destination is reached.

Time considers the user's most frequent location during the current time interval. While Chaoming Song and his colleagues showed that, on average, a user's location coincides with his or her most frequent location for the current hour in a week 70 percent of the time,[1] prediction is most certain during times when the user is least mobile; the majority of users spend most of their day relatively stationary. The periods of movement, which we focus on, are much harder to predict.

Combining these, *OriginTime* and *TwoOriginTime* provide a standard approach to Markovian location prediction. This extension was recommended by Ashbrook and Starner[2] and forms the core of basic location prediction algorithms.

Region-based configurations. We extend these configurations using our regions. As the number of inputs increases, so too does the required training data. Configurations with more than three inputs were prohibitively expensive to train and resulted in overfitting. We propose four configurations, each with two or three inputs.

OriginRegion and *OriginRegionTime* extend the nonregion configurations and utilize the core knowledge that regions capture;

TABLE 3
Model and ensemble results.*

	Model	Precision (%)	Recall (%)	Foresight (%)	Hindsight (%)
Nonregion configurations	Origin	68.38 σ 6.33	02.31 σ 2.00	100.0	—
	TwoOrigin	—	00.00	—	—
	Time	51.79 σ 4.82	27.93 σ 4.39	75.49 σ 4.96	—
	OriginTime	65.25 σ 5.27	06.85 σ 2.19	100.0	—
	TwoOriginTime	64.64 σ 5.90	04.49 σ 2.15	100.0	—
Region-based configurations	OriginRegion	76.56 σ 3.92	55.08 σ 4.69	52.71 σ 4.51	78.67 σ 6.37
	TwoRegion	67.43 σ 3.93	**72.59** σ 4.94	48.15 σ 4.58	**85.21** σ 6.50
	OriginRegionTime	74.58 σ 3.71	65.40 σ 4.23	53.99 σ 4.47	73.88 σ 6.34
	TwoRegionTime	69.32 σ 4.02	68.35 σ 5.17	49.42 σ 4.51	83.41 σ 6.40
Ensembles	EnsembleBest2	76.59 σ 3.79	65.30 σ 4.56	54.15 σ 4.50	76.90 σ 6.25
	FilterBoost	**82.47** σ 3.80	58.93 σ 4.63	48.00 σ 4.72	73.04 σ 6.28

* The maximum values for each metric are in bold, and the standard deviations are included.

boundaries are formed where there is a change in likelihood. Updating at these boundaries allows the model to quickly verify or reject predicted destinations.

TwoRegion and *TwoRegionTime* take an entirely new approach. The model observes which regions the user is leaving and entering as a representation of the user's location and direction. Estimates are updated when, and only when, there is likely to be a meaningful change in likelihood as defined by the region boundaries.

Model ensembles. Observations of the above models showed some characteristics of a weak learner, prompting an investigation into whether ensemble learning could improve performance. We present two examples from this investigation.

Best2Ensemble is a simple aggregator that trains three models on the same training data. In this case, the model configurations used are *Time*, *OriginRegionTime*, and *TwoRegion*. Each model produces an estimate, and the average of the two highest estimates for each destination is taken, increasing confidence without allowing outliers to trigger predictions.

FilterBoost is a basic boosting algorithm that we use to train two *OriginRegionTime* models. The first is trained on the first half of the training data, and then used to predict the remaining training data. Any trip not correctly predicted is added to the training set for the second model. Correctly predicted trips also have a 50 percent chance of being added. Thus, the training data for the second model contains trips more likely to be poorly predicted by the first. Each has an equal vote in the ensemble's final estimate.

Evaluation

For each user, a single region map was used for all tests. Prediction models were trained on the first two-thirds of their trips, then given one location record at a time to produce an estimate—that is, a probability distribution for the likelihood of the user ending the trip at each destination.

A *prediction* was made each time the estimate for a single destination breached the confidence threshold of 90 percent. Estimates vary throughout trips, so any number of predictions might occur during a single trip.

This threshold was chosen for an application that requires high precision to provide only useful interruptions; the threshold would vary by application.

Along with *precision* and *recall*, we defined two additional evaluation metrics:

- *Foresight*: The mean proportion of the trip remaining when the correct prediction is first made and not later withdrawn (that is, how quickly the model recognizes a pattern).
- *Hindsight*: The mean proportion of the trip remaining when an incorrect pretrip prediction (by OriginTime) is rejected (that is, how quickly the model recognizes a change in routine).

These additional metrics provide a more intuitive evaluation of how useful predictions from each model would be.

Results and Discussion

Table 3 shows our results. The results show percentages with standard deviation from simulation of all users in GeoLife, MSR Privacy, and Google Location History sources with more than 120 trips. In all, this is a total of 96 users and 4,032 trips, excluding training data. The maximum values are in bold.

Nonregion configurations show extremely low recall, with *TwoOrigin* failing to make a single prediction. Adding more parameters has little impact and significantly reduces recall. The region representation significantly increases recall by increasing resolution and uncovering useful information. Patterns are stored and recognized, and a larger amount of training data is available. The increased knowledge also improves precision, with almost all region-based configurations outperforming the nonregion configurations.

The *Time* configuration alone has surprisingly strong performance, supporting the findings of Song and his colleagues.[1] However, when combined with other parameters, our results are consistent with previous observations of lower than expected sensitivity to time[11,22] and diminishing returns.[22] Increasing time resolution reduces training data per state and produces a poorer fit.[21] Even if more data can be collected, it is rarely beneficial; overly long training periods resulted in poorer predictions, because the user's mobility patterns change over time.

We introduced two sets of region-based configurations; *OriginRegion* and *TwoRegion*, each with their respective time-added configuration. As expected, *OriginRegion* performs extremely well by adding a progress and direction representation to the model proposed by Ashbrook and Starner.[2] Interestingly, the *TwoRegion* configurations perform almost as well by simply understanding a region boundary and direction. This is a clear demonstration that the regions have captured important changes in knowledge, and that this can be efficiently leveraged to predict movements.

Overall gains from the aggregation ensemble approach appear minor, but it is able to achieve higher precision without letting recall fall below average. Variance is lower than simple region-based models across all metrics, and gains are particularly visible for more difficult users.

FilterBoost has stronger precision, but lacks in recall—a pattern seen across all variations of boosting tested. Training on a subset exaggerates the data sparsity problem for each learner, and a steep drop-off was noticed as the number of learners increased. Common algorithms such as AdaBoost performed poorly for this reason. Interesting results also appear when we observe the foresight and hindsight metrics. Many of the nonregion configurations have 100 percent foresight, indicating that predictions are made *only* before the user leaves a destination. For this reason, we ignore their hindsight.

Even if more data can be collected, it is rarely beneficial; overly long training periods resulted in poorer predictions.

Hindsight is extremely strong through all region-based configurations, particularly in *TwoRegion*, indicating that the model can quickly identify when a user's path doesn't follow the expected route. Turning off a highway is likely to result in crossing a region boundary; if users do this before their regular exit, the model is immediately aware of a change. This knowledge is again generated without the need for a street map and while maximizing region size.

Foresight across all region-based models indicates that predictions are made on average halfway through a user's trip. While 50 percent of the trip would give ample time to anticipate a user's needs, high variance might mean that some predictions are too late to provide practical benefit.

Additional problems were noted with the prediction model used. The use of a stochastic model caused estimates to fluctuate throughout a trip. A model that makes greater use of a user's recent path would enable more accurate estimates. Our model provides a valuable comparison between alternatives but would easily be outperformed by a more advanced, or simply a better suited, prediction model.

The comparison between nonregion- and region-based configurations clearly demonstrates that regions provide additional knowledge to improve predictions. The significant increase in recall reflects the ability to generalize and perform well despite data sparsity. Finally, the stronger than expected performance of the *TwoRegion* configurations indicates that knowledge of the user's movements has been captured by the region boundaries. This result is particularly encouraging, given how few and large the regions are, and it demonstrates the potential for power savings by reducing frequency of operations.

We identified significant potential for future improvements in both region formation and prediction. Adding these improvements to the performance gains observed could allow for a practical and efficient approach to location prediction for low-power devices.

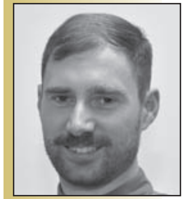
Our results demonstrate the potential of our approach, but they would be equally applicable if regions were formed using alternate algorithms. Reducing the size of regions increases recall slightly, indicating that we have not quite captured all information about a user's movements, and an alternative region formation algorithm might improve on this. Clustering, support vector machines, and entropy monitoring all have the potential to improve region formation.

Although the inclusion of regions improved predictions, the prediction algorithm used was developed only for evaluation and was too naive for a realistic implementation. Regions could be included in alternate prediction algorithms to see if the same gains can be achieved. The inclusion of additional contextual parameters might also improve performance.

Finally, testing on low-power devices is necessary to fully validate and analyze potential energy savings. Testing with low-accuracy data could validate how regions compensate for low location accuracy. If nothing else, a simple automated implementation would stir interest in improving location prediction on low-power devices and truly demonstrate the benefits that regions provide.

the AUTHORS

Ian Craig is a software engineer at Microsoft, but this research was carried out while he was at the University New South Wales (UNSW), Australia. His research interests include developing algorithms to extract knowledge from large location datasets to improve contextual awareness in ubiquitous computing. Craig received a BA in mechatronic engineering, and a BSc in computer science from the UNSW, Australia. Contact him at ian.craig@unswalumni.com.



Mark Whitty is a lecturer in the School of Mechanical and Manufacturing Engineering at University of New South Wales (UNSW), Australia. His research interests include indoor localization, 3D mapping, unmanned ground vehicles, and image processing as applied in field robotics, including yield estimation in viticulture. Whitty received a PhD in mechatronic engineering from UNSW. He is a member of IEEE and is Secretary of the Australasian Robotics and Automation Association. Contact him at m.whitty@unsw.edu.au.

REFERENCES

1. C. Song et al., "Limits of Predictability in Human Mobility," *Science*, vol. 327, no. 5968, 2010, pp. 1018–1021.
2. D. Ashbrook and T. Starner, "Using GPS to Learn Significant Locations and Predict Movement across Multiple Users," *Personal and Ubiquitous Computing*, vol. 7, no. 5, 2003, pp. 275–286.
3. T.M.T. Do et al., "A Probabilistic Kernel Method for Human Mobility Prediction with Smartphones," *Pervasive and Mobile Computing*, vol. 20, issue C, 2014, pp. 13–28.
4. J. Sen, "Ubiquitous Computing: Applications, Challenges and Future Trends," *Embedded Systems and Wireless Technology: Theory and Practical Applications*, CRC Press, 2012, pp. 3–10.
5. A. Krause et al., "Trading off Prediction Accuracy and Power Consumption for Context-Aware Wearable Computing," *Int'l Symp. Wearable Computers* (ISWC), 2005, pp. 20–26.
6. F. Liu et al., "Gearing Resource-Poor Mobile Devices with Powerful Clouds: Architectures, Challenges, and Applications," *IEEE Wireless Comm.*, vol. 20, no. 3, 2013, pp. 2–10.
7. A.Y. Xue et al., "Destination Prediction by Sub-Trajectory Synthesis and Privacy Protection against Such Prediction," *Proc. IEEE 29th Int'l Conf. Data Engineering* (ICDE), 2013, pp. 254–265.
8. L. Liao, D. Fox, and H. Kautz, "Extracting Places and Activities from GPS Traces Using Hierarchical Conditional Random Fields," *Int'l J. Robotics Research*, vol. 26, no. 1, 2007, pp. 119–134.
9. B. Hoh et al., "Preserving Privacy in GPS Traces via Uncertainty-Aware Path Cloaking," *Proc. 14th ACM Conf. Computer and Communications Security* (CCS), 2007, pp. 161–171.
10. J. Alvarez-Garcia et al., "Trip Destination Prediction Based on Past GPS Log Using a Hidden Markov Model," *Expert Systems with Applications*, vol. 37, no. 12, 2010, pp. 8166–8171.
11. R. Simmons, B. Browning, and V. Sadekar, "Learning to Predict Driver Route and Destination Intent," *Proc. IEEE Intelligent Transportation Systems Conf.*, 2006, pp. 127–132.
12. J. Krumm, "Where Will They Turn: Predicting Turn Proportions at Intersections," *Personal and Ubiquitous Computing*, vol. 14, no. 7, 2010, pp. 591–599.
13. L. Liao et al., "Learning and Inferring Transportation Routines," *Artificial Intelligence*, vol. 171, nos. 5-6, 2007, pp. 311– 331.
14. Y. Zheng et al., "Understanding Mobility Based on GPS Data," *Proc. 10th Int'l Conf. Ubiquitous Computing* (UbiComp), 2008, pp. 312–321.
15. Y. Zheng, X. Xie, and W. Ma, "GeoLife: A Collaborative Social Networking Service among User, Location, and Trajectory," *IEEE Data Eng. Bull.*, vol. 49, no. 2, 2010, pp. 32–39.
16. Y. Zheng et al., "Mining Interesting Locations and Travel Sequences from GPS Trajectories," *Proc. 18th Int'l Conf. World Wide Web* (WWW), 2009, pp. 791–800.
17. J. Krumm and A. Brush, "MSR GPS Privacy Dataset 2009," Microsoft Research, 2009; http://research.microsoft.com/ jckrumm/GPSData2009.
18. N. Marmasse and C. Schmandt, "Location-Aware Information Delivery with ComMotion," *Proc. Int'l Symp. Handheld and Ubiquitous Computing*, 2000, pp. 157–171.
19. N. Brouwers and M. Woehrle, "Detecting Dwelling in Urban Environments Using GPS, WiFi, and Geolocation Measurements," *Proc. 2nd Int'l Workshop on Sensing*, 2011, pp. 1–5.
20. R. Hariharan and K. Toyama, "Project Lachesis: Parsing and Modeling Location Histories," *Geographic Information Science*, LNCS 3234, 2004, pp. 106–124.
21. A. Stolcke and S.M. Omohundro, "Hidden Markov Model Induction by Bayesian Model Merging," S.J. Hanson, J.D. Cowan, and C.L. Giles, eds., *Advances in Neural Information Processing Systems*, Morgan-Kaufmann, 1993, pp. 11–18.
22. J. Krumm, *A Markov Model for Driver Turn Prediction*, tech. report, SAE Int'l, 2008.