When Will You Be at the Office? Predicting Future Locations and Times

Ingrid Burbey and Thomas L. Martin

Bradley Department of Electrical and Computer Engineering
Virginia Tech, Blacksburg, VA
{iburbey,tlmartin}@vt.edu

Abstract. The purpose of this paper is to predict people's future locations or when they will be at given locations. These predictions support proactive, context-aware and social applications. Markov models have been shown to be effective predictors of someone's *next* location [1]. This paper incorporates temporal information in order to predict *future* locations or the times when someone will be at a given location. Previous models use sequences of location symbols and apply Markov-based algorithms to predict the next location symbol. In our model, we embed temporal information within the sequence of location symbols. To predict a future location, we use the temporal information as the previous state (or context) in the Markov model to predict the location that is most likely at that given time. To predict *when* someone will be at a location, we use the location as the context and predict the time(s) the person will be at that location. The model produces up to 91% accuracy for predicting locations, and less than 10% accuracy for predicting times. We show that prediction of location and prediction of time are two very different problems, because the number of predictions produced by the Markov model differ greatly between the two variables. A heuristic algorithm is proposed which incorporates additional context to improve predictions of future times to 43%.

Keywords: location-prediction; time-prediction.

1 Introduction

One of the goals of ubiquitous computing is context-awareness, enabling our devices to become our invisible assistants instead of one more device to be managed. Our proactive devices need to "infer the intent of the user" [2]. Knowing the location of a device (and therefore the location of the owner) contributes to knowing the context and the activity of the user. The ability to *predict* the location of the user can support context-aware applications such as assistive devices, reminder systems, social networking applications, and recommender systems.

The ability to predict *when* someone will be at a given location can also support context-awareness. This knowledge can help friends rendezvous or coworkers meet. It can detect anomalies, for example, sending an alert after discovering that someone did not arrive when expected. Collaboration between co-workers can be coordinated when one knows the temporal routines of the other.

This paper examines how to predict future *locations*, such as where someone will be at a given time. In addition, we examine how to predict future *times*, such as when someone will be at a given location. We embed temporal information into location sequences and begin by using a Markov model to predict symbols in the sequence. We then improve upon the Markov temporal predictions with a heuristic model which uses more context than is available in the Markov model.

Our experiments were run on data collected from PDAs which were toted by freshman on the UCSD campus in 2002 [3]. We parsed the data using two levels of temporal granularity: 1-minute timestamps and 10-minute timestamps. One-minute timestamps reflect movement, while the 10-minute granularity focuses on our significant locations or destinations. The experimental results show that a first-order Markov model with fallback can predict future locations with an accuracy of up to 91% for the 10-minute timestamps and 85% for the 1-minute timeslots.

Predictions of future times, such as when someone will be at a given location, are not as successful. Experimental results with the same data return a prediction accuracy of less than 10% unless the model tests against all of the predictions returned, both good and bad, which often (85% of the time) covers the entire 24-hour day. A heuristic algorithm is then used which can incorporate further context, such as a time and/or location that was observed earlier in the day. The heuristic algorithm improves the prediction of future times up to 43% accuracy.

The remainder of this paper is organized as follows. Section 2 discusses several applications for location and time predictions. Section 3 reviews related work in predicting next locations and/or future times. Section 4 reviews the Markov Model. Section 5 contains the details of the data, the experiments using both the Markov model and the heuristic model, and the results. Section 6 summarizes and concludes the paper.

2 Applications

Location prediction can inform many types of proactive, context-aware applications. For a single user, location prediction can provide reminders and recommendations. When predictions are shared with others, social networking can be enhanced, assistive support can be given, resources can be reserved and advertising can be targeted. This section discusses each of these applications in further detail.

For a single user, location prediction can be used for reminders and recommendations. A simple example is a reminder to pack up library books in the morning on a day you are expected to go past the library. The Magitti Recommendation System [4] remembers its owner's and other's past activities to recommend activities that the owner may enjoy at her current location. With predictive capability, a recommendation system such as Magitti could additionally recommend activities for future locations and remind one what to bring.

Predictive, proactive devices can provide 'cognitive assistance' [5] to those who may need help due to mental disability, aging or simple distraction. Predictions can be used to detect anomalies, such as when a user deviates from the expected path or routine [6].

The "Opportunity Knocks" system [5] tracks a user as he rides the bus home to his destination. If he accidentally boards the wrong bus, or gets off at the wrong stop, the system notices the discrepancy, politely 'knocks' to alert the user, and directs him on where to go to catch the correct bus. The current implementation of the system asks the user to identify his destination before he boards the bus. With predictive ability, the application could either predict the user's final destination or suggest a reduced set of likely destinations for the user to select.

Sharing predictions with a caregiver can promote independence. In [7], Chang, Liu and Wang developed a social networking system to assist people with mental illness. The system includes alarms for situations where the user does not arrive at her expected destination at the expected time or if she deviates from her expected path home. Predictions of future destinations could inform assistive technology applications such as these.

Social networking can also be enhanced using predictions. When predictions are shared, a friend can determine when to 'run-into' or serendipitously rendezvous with someone (or avoid them). Common destinations could be used for social match-making [8]. Merchants could use predictions to target advertising, perhaps offering a coupon to lure you to a different restaurant for lunch than your usual stop.

Privacy can be supported by location-prediction if it replaces tracking. For example, a corporation that tracks employees in order to find the closest technician in case of a problem can conceivably use location-prediction instead to produce a list of technicians who are likely to be nearby.

Ashbrook and Starner, in [9], suggest several multiple-user applications of location information in addition to those already mentioned. One application is the exchange of favors, such as errands that someone else could possibly do in your place. Another application could support coordinating a meeting of several people.

Computer-supported-collaborative-work (CSCW) is an important area of research, especially with the trend toward remote work. When co-workers are co-located, they learn each other's routines and availability. This ambient information gets lost when workers are in different locations. Begole, Tang and Hill [10] developed algorithms to predict when office workers would be online, indicating that they were in the office and available for communication. This is an application of predicting *time*, such as when someone arrives in the office in the morning or returns from lunch.

3 Related Work

Prediction of the next cell in a sequence of locations has been investigated in the networking and communications arenas in order to improve resource reservation and quality of service [11-17]. These efforts focus on predicting the next cell and do not concern locations that will occur further out into the future. Smart Homes predict in order to reduce paging messages and allocate resources, such as light or heat in the next room the occupant will enter [18-21]. These experiments use a limited location space and are domain-dependent in that they can use knowledge of the geometry of the network or the home in order to constraint and improve the algorithm. In addition, these works can assume continuous location updates. The inhabitants of a Smart Home do not leave the kitchen and appear in the bedroom without walking down the hallway.

In [22], Zeibart et al. successfully predict driving destinations given a partially traveled route. Our goal of pedestrian destination prediction is similar. However, we cannot assume that location updates are continuous in time. Locative devices, such as our cell phones, will occasionally be turned off, get left behind or enter areas where location systems do not have coverage, such as GPS ‘urban canyons.’ Applications such as Facebook [23] and Twitter [24] rely on user updates, which mean that there will be significant time lapses between location updates. In addition, to support privacy concerns, any location-logging application must allow the user to delete records or stop recording if the user desires [25-27].

The authors of [28] use a combination of a Markov model and a heuristic model to predict the number of users at each base station in a cellular network. They refer to the single best prediction returned as the ‘hard decision’ and the set of all possible predicted locations as the ‘soft decision.’ In our work on predicting future times, we also consider hard versus soft decisions.

4 Markov Models for Prediction

Popular data-compression algorithms, such as Lempel-Ziv (used in PKZIP and gzip), use probabilistic models based on Markov theory. These algorithms use historical data to predict the next symbol in a sequence. When compressing, these algorithms use the least number of bits to encode the most predicted symbols or sequences of symbols in order to reduce the size of the output. These algorithms can also be applied to prediction. Begleiter et al. applied several variable-order Markov models to prediction tasks [29]. One of their test applications was the prediction of music, which was of interest to us because the representation of music includes notes, their starting times and durations, which is similar to the temporal information we embed in our location sequences. They achieved good results with the Prediction-by-Partial-Match (PPM) algorithm, which is the model we chose to apply to our problem. One advantage of the PPM algorithm is that it falls back to lower-orders automatically, which incorporates the findings of Song et al., that a low-order Markov model with fallback was the best predictor for location sequences [1]. Cleary et al. provide an excellent tutorial on using the PPM model for prediction in [30].

5 Method

This section of the paper describes the raw data used, the pre-processing steps, modifications made to the Markov model, and the implementation of the heuristic model.

5.1 Location Using 802.11 Access Points

IEEE 802.11 access points can be used as location beacons [31-33]. They are ubiquitous in urban and suburban areas and many mobile devices include built-in

IEEE 802.11 wireless capability, making IEEE 802.11 an inexpensive location-determining technology. Currently, our requirement is room-level or building-level location granularity, which is well supported by available IEEE 802.11-based location systems. In this project, we use the Access Point ID as the location label. We do not take the significant step of converting the access point ID into a useful place name, which in itself is a difficult problem [34, 35].

As part of the Wireless Topology Discovery (WTD) project at UCSD, researchers issued PDAs to 300 freshmen and collected WiFi access traces [3]. The data were collected during an 11 week trace period during the fall of 2002.

While a student's device was powered on, WTD polled the IEEE 802.11 wireless status every 20 seconds. It recorded a timestamp, access point (AP) ID, signal strength, whether or not the device was using AC power and whether or not the device was associated with the access point. Access points that were sensed, but not associated, were also recorded. Later, we used the sensed access points to create a list of access points that were neighbors of the associated access points and were likely to be in close proximity.

5.2 Preprocessing

The raw data from the WTD experiment combines all logs from all users into one file. Our first step was to divide the raw data into individual files for each user. Records that had the same time and date stamp were compared and the record with the highest signal strength was retained and any other records (usually of sensed, non-associated, access points) were discarded. Contiguous records were combined, where contiguous is defined as records with the same user number, the same access point and where the starting time was within one minute of the previous record's starting time. (Recall that polls are approximately 20 seconds apart.) The duration of each session was calculated and included in the output. These combined sessions were not allowed to run over a one day boundary. (In reality, there were no sessions over 20 hours long.)

We wanted two different views of the data. The first is movement data, which represents the information that would be recorded by an always-on mobile device. The second is destination data, which represents the locations that a user might disclose using a social networking application. These locations are the *significant locations* in someone's day. In this work, significant locations are determined solely by a length of stay of at least 10 minutes [36]. In the future, this definition may expand to include recurrence [35]. Throughout the rest of this paper, we label the movement data as "MoveLoc" (for "Movement Locations") and the destination data is labeled "SigLoc" (for "Significant Locations").

The MoveLoc Dataset. The starting time timestamps in the MoveLoc dataset are rounded to the closest preceding minute. We wanted the MoveLoc dataset to include as much movement data as possible, so all sessions, even those with durations of less than 20 seconds, were included. If more than one session occurred in the same one-minute window, then the session with the longest duration was used for that one-minute timeslot and the others were discarded.

Fig. 1 shows the MoveLoc data for User 3. The color bar shows that this user visited 30 locations over the 10 week recording period. Time-of-day is along the x-axis. One can observe that this user had some regular locations between noon and 2:00pm, and some of the movement between locations is indicated by the change in colors/shades.

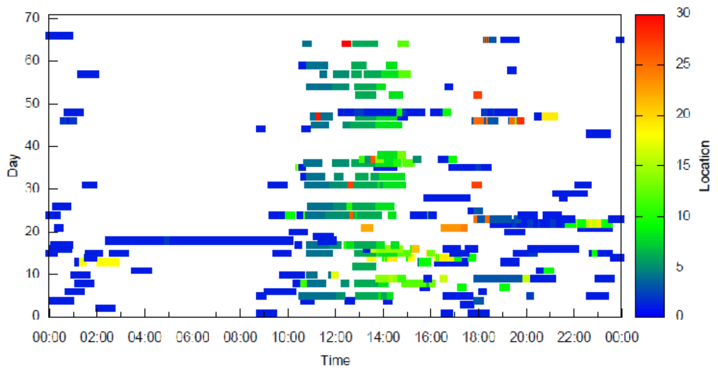

Fig. 1. User 3's MoveLoc (1-minute or less) data

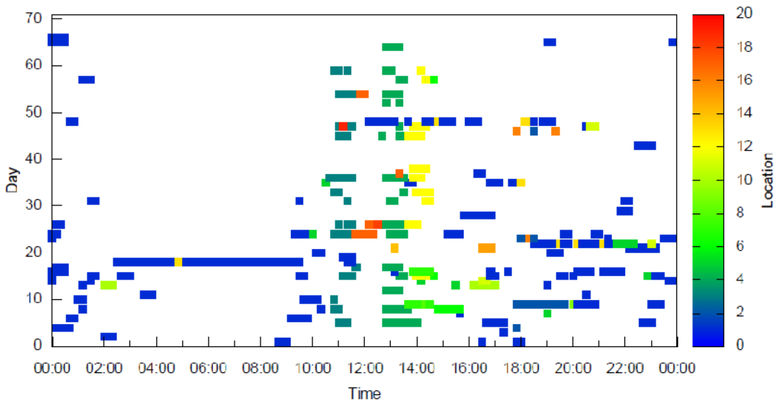

Fig. 2. User 3's SigLoc (10-minute) data

The SigLoc Dataset. The SigLoc dataset is comprised of places where the user spent at least 10 minutes. The first step in creating this view was to remove all sessions with durations of less than 10 minutes. The starting times of the remaining sessions were rounded down to the closest 10 minutes. The SigLoc data for User 3 is shown in Fig. 2. One can observe that the number of locations dropped from 30 to 20 and the transitions between locations have been removed. (Please note that the colors or shades assigned to various locations are not the same for the two figures.)

Finally, the MoveLoc and SigLoc data were then stored in two formats: 16-bit symbols for the Markov model and a human-readable form for the SEQ model.

5.3 PPM for Predicting Location

A PPM (Prediction-by-Partial-Match) implementation of a variable-order Markov Model was used as the basis for the prediction model. Source code from [37] was used as the basis of our implementation. Due to the large number of possible timeslots, the data were pre-processed to encode times and locations as 16-bit symbols, with one range for times and another for locations. It is not necessary to restrict the symbols to a given range; however it is useful for validating the results returned by the model.

When using the model to prediction future locations, the data are represented as a sequence of (time, location) pairs. Each user's data are in a separate file, which has the format:

$$date; \{time_0, location_0, time_1, location_1, \ldots time_n, location_n\}$$

Ten weeks of data are partitioned into test sets, or runs, where each run consists of five weeks of training data and one week of testing data, based on earlier tests [38]. For example, one run could consist of User 3's data from weeks 2 – 6 for training and User 3's data from week 7 for testing.

The model is trained for the first order. (A similar experiment using third order is reported in [38]). Once the model is trained on five weeks of data, the following week is used to test the model. The test data are parsed into individual (time, location) pairs. For each pair, the time is fed into the trained model as the input (context), and the model returns a list of predicted locations with their probabilities, sorted with the highest probability predictions at the top of the list. The model then checks each of the returned predictions against the test location, which is the correct answer. The prediction(s) with the highest probability are checked first (and labeled "Best" in the figures of results.) The lower probability predictions are checked next (and are included in the "all predictions" category in Fig. 3).

If none of the returned predictions are correct, the neighboring access points of the incorrect predictions are checked. Neighboring access points were determined during preprocessing when a list is created of all of the access points which were sensed at the same time as each associated access point. These access points are likely to be in the same building. For example, if, in our test, the correct answer was access point #72, and the Markov model returned a prediction of access point #63, the model would check to see if access point #63 was a neighbor of #72 and likely to be in the same proximity.

If the input time was not found at the first order, the PPM algorithm falls back to the 0th order and returns a list of the most likely locations, regardless of the input time. This means that the model has no information about the time that was entered as the context, so it simply returns the most popular locations.

The results of using the Markov model to predict future locations are shown in Fig. 3. The lowest results are for the MoveLoc dataset which uses 1-minute timeslots. Using 1-minute timeslots increases the number of states in the Markov model up to tenfold compared to 10-minute timeslots. This state explosion increases the number of possible predictions and decreases the probability for each, leading to worse results. For example, if the user arrives at location *x* at 8:00am and the next day arrives at 8:01am, these times are recorded as separate states in the Markov model, which reduces their individual probabilities. In the SigLoc dataset, which uses wider 10-minute windows, these two records fall into the same 8:00am state, increasing its count, and therefore its probability. The increased size of the timeslot compensates for variability in arrival times by using a longer window of time that is considered to be a correct prediction.

The best results, 91% for the SigLoc data, are achieved when all of the returned predictions are used, regardless of their probabilities. The median number of predictions returned is 2, with a maximum of 13 for the MoveLoc data and 9 for the SigLoc data.

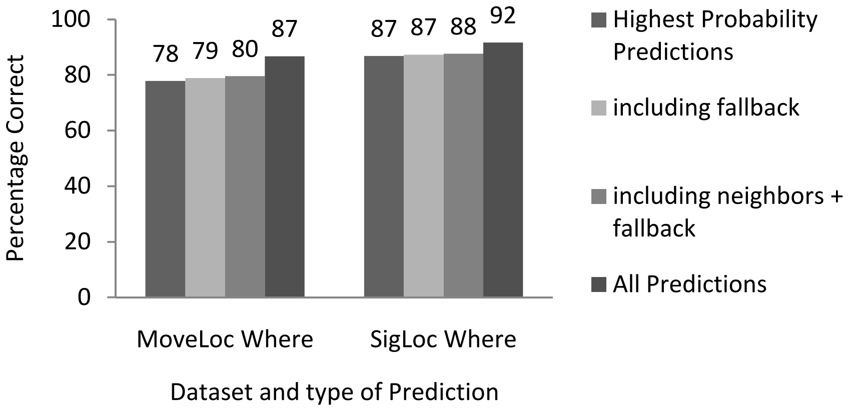

Fig. 3. Predicting Future Locations using the Markov Model

5.4 PPM for Predicting Time

We now turn to the converse problem. Instead of asking questions about someone's future location, such as "Where will Bob be at 10:00am?" we ask questions about *when* someone will be at a given location, such as, "When will Bob be in his office?"

We began with the identical Markov Model. Instead of training on data that was formatted as a sequence of times and their corresponding locations,

$$\{time_0, location_0, time_1, location_1, \ldots time_n, location_n\},$$

the sequences are rearranged to form (location, time) pairs:

$$\{location_0, time_0, location_1, time_1, \dots location_n, time_n,\}$$

where $location_t$ is the context, or the corresponding location, for $time_t$.

The model was trained and tested with the same runs of five-weeks of training data with a corresponding week of testing data.

Fallback was not used in this model because it does not make logical sense. In the location application of the model, which is answering the 'where' question, it may make sense for an application to fall back to the most common location. For the 'when' question, such as "When will Bob be in Hawaii," it does not make sense for the model to respond, "I have no data for 'Hawaii,' but the most probable time is 10:00am, so I will predict that Bob will be in Hawaii at 10:00am. In such cases, the model should refuse to predict.

If none of the time predictions returned were correct, they were then checked to see if they were within 10 or 20 minutes of the correct time. This corresponds to the checking of neighboring access points that was done for location prediction.

The results for using the Markov model to predict the time someone will be at a given location are shown in Fig. 4. Initial results are dismal: less than 10% of the best predictions are correct for both datasets. Results improve slightly by allowing predictions to be within 10 or 20 minutes of the correct time.

The high value achieved by using all of the predictions returned comes at a price – the returned predictions cover all times of the day! In other words, the model is asked, "When will Bob be at location *x*?" and the model responds, "Sometime today between midnight last night and midnight tonight." This result is useless for an application.

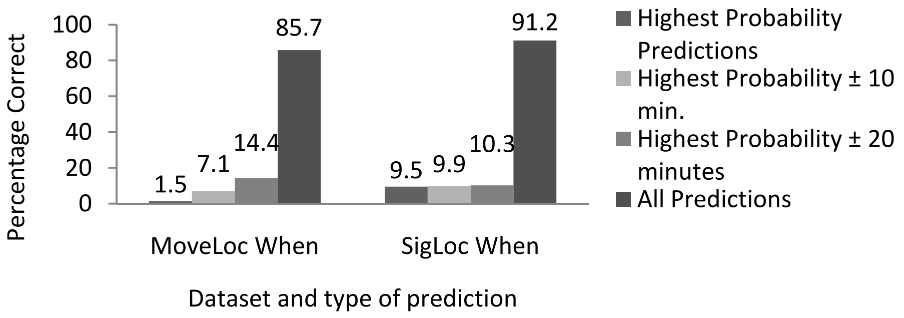

Fig. 4. Predicting Times at Future Locations using the Markov Model

The reason for the differences in our location and time prediction results are apparent when we look at the quantity of predictions returned. The upper bound on the number of predictions returned is the maximum number of states. For time predictions, the maximum number of states is 144 for the 10-minute, SigLoc data, and

1,440 for the 1-minute, MoveLoc data. The maximum number of locations varies by user. Table 1 shows the median number of unique locations and times in the training files. The number of location states is much less than the number of temporal states; hence, the model is more likely to get the location correct. The numbers of predictions returned listed in Table 2 also reflect this discrepancy. The number of predictions returned for location prediction is much less than the number of predictions returned for time prediction.

Table 1. Average Number of Unique Locations and Times for each Training File

Dataset	Ave. # Locations / Training File	Ave. # Times / Training File
MoveLoc	18	646
SigLoc	5	68

Table 2. Number of Predictions Returned for each Test

DataSet	Question	Median # Predictions returned	Max. # Predictions returned
MoveLoc	Where?	2	13
SigLoc	Where?	2	9
MoveLoc	When?	1439	1440
SigLoc	When?	144	144

Recall that the model returns *all* of the predictions found for a given context. The results for "all predictions" in Fig. 4 use all of the predictions returned, which is most cases covers the entire day. Consider Fig. 5, which graphically shows the set of predictions returned for a hypothetical query asking when Alice will be at the office. The graph shows that Alice usually arrives around 9:00am, goes to lunch around noon, and leaves for the day at 5:00pm. The highest probability prediction is the single prediction at 9:00am, which corresponds to the lowest results in Fig. 4. This type of result is useful for applications which need to know the singular times of the day when someone is going to be in a given location. Using all of the reported predictions corresponds to the best results from Fig. 4. These results might be useful to an application that needs to know all of the possible times that Alice might be in her office. However, if Alice left her mobile device in the office overnight one time, these results would always cover the entire day because there would be at least one count for that location at every possible timestamp.

Thresholds. Fig. 5 shows some thresholds that might be used to determine a subset of predictions to be considered. For example, the threshold at 90% would return the prediction that Alice would be likely to be in the office from 8:30am until noon and from 1:10pm until 4:40pm. This broader prediction could be more useful to an application than the single prediction of 9:00am that is returned with a tighter threshold of 0% (which returns the highest probability predictions only).

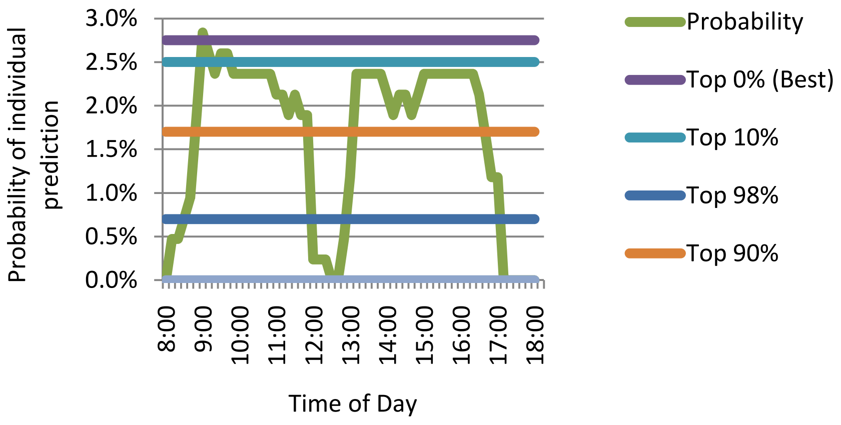

Fig. 5. Thresholds for a Hypothetical set of Predictions

Thresholding was then implemented in the model. Recall that the model returns a list of predictions, each labeled with its probability. The predictions are sorted in order of probability, from the highest probability to the lowest. To invoke a threshold, the top predictions are checked for correctness until their cumulative probability surpasses the probability threshold. For example, a threshold of 10% uses the top predictions until their cumulative probability is greater than 10%. A threshold of 0% uses only the most likely prediction, which is the same as the 'best' predictions in the Fig. 4. A threshold of 100% uses all of the predictions returned, which is the equivalent of the using all of the probabilities, again as shown in Fig. 4. Fig. 6 and Fig. **7** illustrate the results of using a probability threshold for the MoveLoc and SigLoc datasets.

Observing the graphs in Fig. 6 and Fig. **7**, one can see there is no apparent optimal threshold value. The value of the threshold is dependent upon the goals of the application. One application may want only predictions with a high probability while another application may want to present all of the possibilities to the user. Imprecise knowledge of user intent may still be useful information [2].

Currently, we do not consider the predictions themselves, only their probabilities. In the future, we may consider combining contiguous predictions into a window of time. For example, if the returned predictions include {8:00am, 8:10am, 8:20am,…,10:00am}, they could be combined into one prediction of 8:00am – 10:00am. Consideration needs to be made about how to compute the resulting probability when contiguous predictions are combined. If Bob only occasionally arrives at the office at 8:00am but he is almost always there at 9:00am, how is that information presented to the user? Should the less-likely 8:00am prediction be included in the results presented to the user? A proper visualization of the results can illustrate these probabilities, either with a temperature graph which uses different colors for different probabilities or a graph similar to Fig. 5.

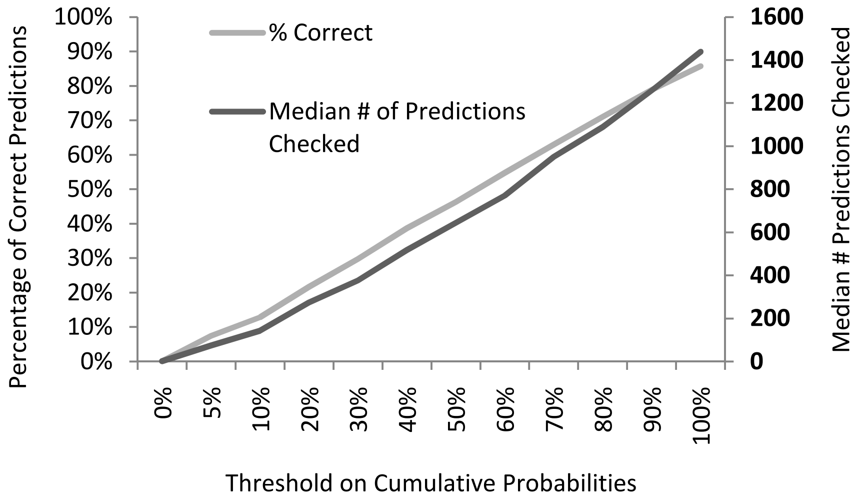

Fig. 6. Applying a Probability Threshold to Time Predictions for the MoveLoc (1-minute) data

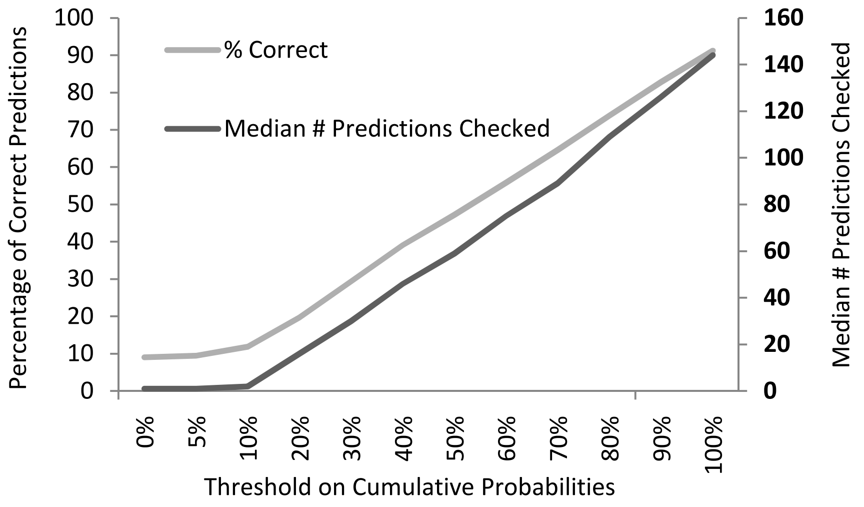

Fig. 7. Applying a Probability Threshold to Time Predictions for the SigLoc (10-minute) data

In addition, when we consider only the probabilities, we are losing some information about the predictions. The PPM model stores counts of the number of times it encounters each pattern in the sequence. Currently, we are testing only against the resulting probabilities and not the counts themselves. The counts could be used to develop a confidence measure about the predictions. The model can return the same probability for two different states, even if it has only one occurrence of the first state and several hundred for the second state. Consider the hypothetical situation where the data has recorded Bob at the mall once at 10:00am, but over the course of 10 weeks, has recorded Bob at the office at all different times between 8:00am and 5:00pm. If the Markov model is queried as to when Bob will be at the mall, it will return a prediction of 10:00am with a probability of 100%. If the same model is queried as to when Bob will be in the office, it will return possibly hundreds of predicted times, and each could have a probability of less than 1%. At first glance, it could appear that the Markov model is more confident that Bob will be at the mall at 10:00am, and that conclusion is erroneous. Displaying confidence increases the user's trust in an application [39]. The counts calculated by the model can be used to determine confidence.

5.5 The Heuristic Model: Traversing the Sequence

Even though the Markov model can be used to predict future times, it is not a suitable solution. Too many predictions are returned and there is a loss of contextual information in statistical models such as the Markov model that are based on counting events. We therefore developed an algorithm which retains more contextual information. We call this model 'Traversing the Sequence' or the SEQ model, because it keeps the entire sequence of times and locations intact and, given contextual information such as a previous time or location, it traverse the sequences to predict a future time at a given location.

The SEQ model represents the user's day as a sequence of (arrival time, location) pairs. Note that this is a different representation than the Markov model. In the Markov model, for example, if a user spends 30 minutes in one location, the result is three records in the SigLoc dataset, one for each 10-minute timeslot. In the SEQ model, this location is represented as one pair with the arrival time and the location and no duration information.

The sequence model is an *unsupervised learning* model, in that the model is not trained on historical data that are labeled with the correct answers. The SEQ model stores the historical data and uses it without training to determine future predictions.

Let us illustrate with a simple example. Our user, Bob, usually goes to a coffee shop at 8:00am, arrives at the office at 9:00am and leaves to go home at 5:00pm. Twice during the last ten days, he skipped the coffee shop and arrived at the office at 8:00am in order to prepare for a 9:00am meeting at another location.

The SEQ model stores these daily sequences without modification. When the model is queried as to when Bob will arrive at the office, it searches each of the sequences to find when Bob arrived at the office and finds that on 8 days he arrived at 9:00am and twice, he arrived at 8:00am. The model would return two predictions: 9:00am with an 80% probability and 8:00am with a 20% probability.

Advantages of the SEQ Model. Because the SEQ model retains historical information, queries with additional context can be asked, such as, "When will Bob be in the office if I just saw him at the coffee shop?" In this case, the model would not search any daily sequences that did not include the coffee shop, and would return a prediction that Bob would arrive at the office at 9:00am with 100% probability. The model can be further expanded to include an earlier time as context (e.g. "When will Bob be at the office if I saw him on the road at 8:15?") or both an earlier time and location.

The model can also be expanded to use other forms of context. If the sequences include a day-of-the-week label, the model can search only the sequences that fall on a particular day of the week or a subset of days (e.g. "When will Bob be in the office on the weekend?")

The SEQ model stays current without retraining. The model can be configured to use only the latest data and ignore outdated sequences. The Markov model used previously requires periodic retraining to incorporate recent data.

The model will refuse to predict if it is given context that it cannot find in a sequence. For example, it will refuse to predict if it is queried about a location it has never seen (e.g. "When will Bob be in Hawaii?") or a previous context that has not been encountered (e.g. "It's 7:00pm, when will Bob be in the office today?")

The SEQ model Experiment. The same raw user data that were used for the Markov model were used for the SEQ model. Instead of using 16-bit symbols to encode time and location as was done for the Markov model, the data for the SEQ model are stored in human-readable, comma-delimited files. The SEQ model was written in Python because of its rapid-development support and ease of handling text strings.

The model was tested with different types of previous context, which we refer to as 'order'. In the 0^{th} order test, the only input is the location. The model is queried as to when the user will be at the test location. This location is referred to as the *quest location*. A previous time can be given as additional context; this is noted as '1-time' order. A previous location can also be used as additional context; this is noted as '1-loc' in the model. And finally, both a previous time and location can be used as additional context, creating the second order.

The additional historical context, such as a previous time, place or both, is used to truncate the daily sequences searched. If a previous time and/or location is given, the index of that time and/or location is used as the left edge of the sequence to be searched for the location in question. The pseudo-code for our SEQ model if given additional context of time (context_time) is listed in Fig. 8.

SEQ Model for Predicting WHEN Someone Will Arrive at the Quest_Location
(" When will the user be at the quest_location is she was observed at context_time earlier that day?")

```
Reset counters.
For each daily sequence:
Is the quest_location in the sequence?
If not, increment the count for "Never" and continue onto
the next sequence.
Search the sequence to find the time closest to (equal or
earlier than) the context_time.
If the context time is not found,   increment the count
for "Never" and continue onto the next sequence.
The location of the context_time becomes the left border
of the sub-sequence to be searched.
Search the sub-sequence to find the quest_location.
If not found, increment the count for "Never" and
continue onto the next sequence.
The location of the quest_location becomes the right
border of the sub-sequence to be searched.
Search the sub-sequence to find the context_location.
If not found, increment the count for "Never" and
continue onto the next sequence.
Use the time corresponding to the quest_location as the
predicted time and add it to the counters.
Return the counters.
```

Fig. 8. Pseudo-code for the SEQ Model

Results for the MoveLoc (1-minute) and SigLoc (10-minute) datasets are shown in Fig. 9 and Fig. 10, respectively. The prediction accuracy has improved over the Markov model. Allowing for predictions within 10 or 20 minutes of the correct testing time yields an even higher predictive accuracy.

There are a few causes for the improvement in prediction. First, the SEQ model predicts only arrival times. The Markov model indirectly encodes duration, in that records are repeated in the sequence until the user changes location. The Markov data, therefore, has a chain of multiple records if a user is at a particular location for a length of time. Since the length of the user's stay may vary, the tests on the records at the end of the chain may produce wrong predictions, reducing the average predictive accuracy. (For example, predictions about 5:00pm, when the last arrival time recorded was at 2:00pm, would be incorrect in the current SEQ model.)

The predictions worsen with additional context (the 1loc, 1time and 2 areas of the graphs). There are fewer sequences which match the given context. In some cases, there are no historical sequences that match the additional context and no predictions are made.

The SEQ model uses a smaller dataset of arrival times and locations instead of polled timeslots and locations and returns fewer predictions. This reduction in the number of states improves its performance over the Markov model.

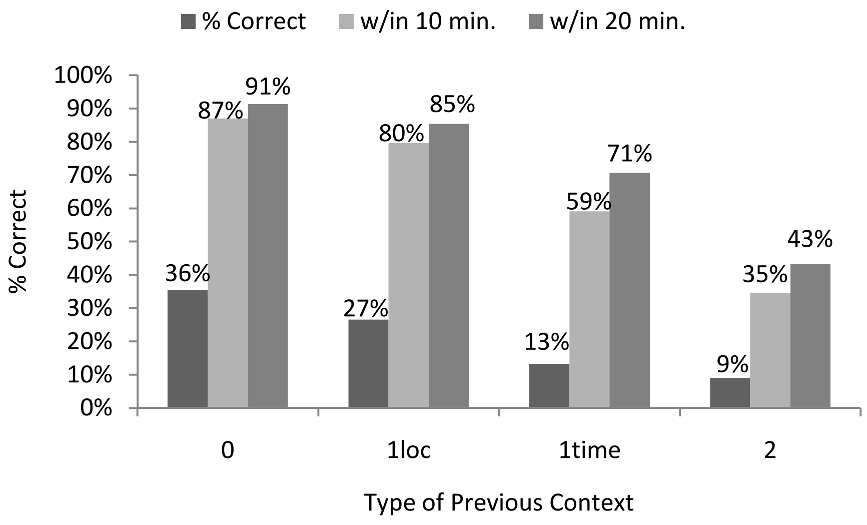

Fig. 9. Results of the SEQ Model on the MoveLoc (1-minute) data

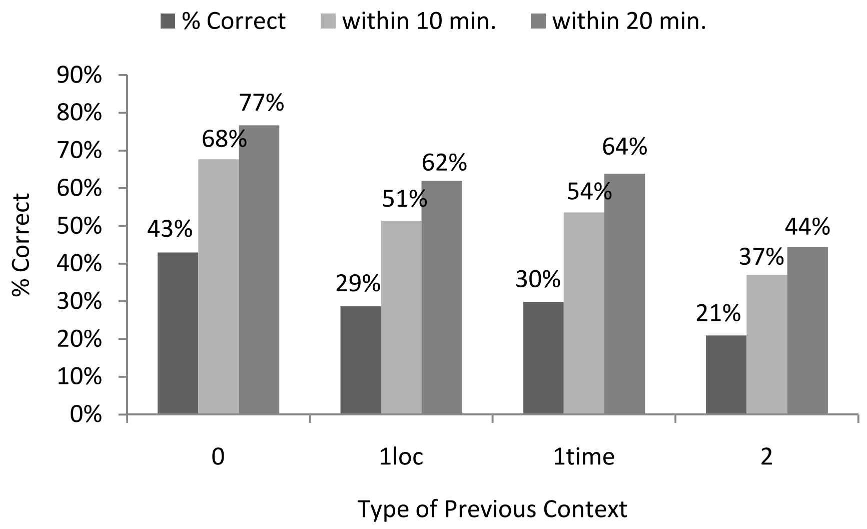

Fig. 10. Results of the SEQ Model on the SigLoc (10-minute) data

6 Summary and Conclusion

Our goal is to develop models for predicting people's future locations and times. This problem is different than previous works because we are not limiting the predictions to the prediction of the *next* location or time, but instead predicting locations and times out into the *future*.

We used data collected from PDAs, and used IEEE 802.11 access point IDs for determining location. We represented the data in two formats: one which represented moving data and used 1-minute timeslots, and one which represented destinations or significant locations and used 10-minute timeslots.

We found that a Markov model with fallback, such as PPM, can predict future locations with accuracy up to 91%. However, the same implementation failed to predict future times. A heuristic model, called the SEQ model, which traversed daily sequences of time and location data, improved prediction of time up to 77%. The heuristic model also supports additional context, such as previous times and/or locations. It can easily be expanded to support other contextual information such as day-of-the-week, or any other information that can be appended to the daily sequences.

We have found that prediction of time is a more difficult problem than prediction of location due to the large number of possible states. Sequence predictors that are effective for predicting next or future locations are not suited for predicting future times.

6.1 Future Work

There are many areas for future improvements in this work. The Hidden Markov Model (HMM) may improve the results of location prediction, as the hidden states could compensate for the unobserved locations between destinations or for situations where two access point labels identify the same location. Visualization of the returned information should be done to impart not just the predicted time or location to the user, but also the relative probabilities. Confidence measures should also be included in the visualization. Visualization becomes more complex when predictions from multiple users are combined.

The SEQ model currently models only arrival time at each location. It can be expanded to include duration information. Duration information would support more useful applications. Prediction of arrival times tells *when* someone will arrive, but it may be more useful to know how long someone will be at a given location. Previous work in learning relative time between events was done in [40].

6.2 Conclusion

This paper is about predicting the future, specifically when someone will be at a given location, or where they will be at a given time. Prediction is a sub-case of context-awareness. In this case, we are not attempting to determine someone's current context or activities, but instead, we take a first step towards predicting future context by predicting future locations or future times at a given location.

We see many applications for such predictions, especially when they are shared among friends, co-workers or care-givers. These predictions can replace the ambient knowledge of our friends and co-workers routines that we pick up automatically if we live or work within close proximity of them. This information can help care-givers remotely and unobtrusively monitor their clients, help friends meet, assist co-workers with contacting others. Hopefully, this work is one small contribution towards the invisible, proactive, assistive devices that are part of the vision of pervasive and ubiquitous computing.

Acknowledgements. The authors would like to thank the anonymous reviewers for their insightful and helpful suggestions.

References

1. Song, L., Kotz, D., Jain, R., He, X.: Evaluating location predictors with extensive Wi-Fi mobility data. In: Proc. INFOCOMM, pp. 1414–1424. IEEE, Hong Kong (2004)
2. Satyanarayanan, M.: Pervasive computing: vision and challenges. IEEE Personal Communications 8, 10–17 (2001)
3. UCSD Wireless Topology Discovery Trace, `http://sysnet.ucsd.edu/wtd/`
4. Bellotti, V., Begole, B., Chi, E.H., Ducheneaut, N., Fang, J., Isaacs, E., King, T., Newman, M.W., Partridge, K., Price, B., Rasmussen, P., Roberts, M., Schiano, D.J., Walendowski, A.: Activity-based serendipitous recommendations with the Magitti mobile leisure guide. In: Proc. CHI. ACM, Florence (2008)
5. Patterson, D.J., Liao, L., Gajos, K., Collier, M., Livic, N., Olson, K., Wang, S., Fox, D., Kautz, H.: Opportunity Knocks: A System to Provide Cognitive Assistance with Transportation Services. In: Davies, N., Mynatt, E.D., Siio, I. (eds.) UbiComp 2004. LNCS, vol. 3205, pp. 433–450. Springer, Heidelberg (2004)
6. Estrin, D., Chandy, K.M., Young, R.M., Smarr, L., Odlyzko, A., Clark, D., Reding, V., Ishida, T., Sharma, S., Cerf, V.G., Lzle, U., Barroso, L.A., Mulligan, G., Hooke, A., Elliott, C.: Internet Predictions. IEEE Internet Computing 14, 12–42 (2010)
7. Chang, Y.-J., Liu, H.-H., Wang, T.-Y.: Mobile social networks as quality of life technology for people with severe mental illness. IEEE Wireless Communications 16, 34–40 (2009)
8. Axup, J., Viller, S., MacColl, I., Cooper, R.: Lo-Fi Matchmaking: A Study of Social Pairing for Backpackers. In: Dourish, P., Friday, A. (eds.) UbiComp 2006. LNCS, vol. 4206, pp. 351–368. Springer, Heidelberg (2006)
9. Ashbrook, D., Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing 7, 275–286 (2003)
10. Begole, J.B., Tang, J.C., Hill, R.: Rhythm modeling, visualizations and applications. In: Proc. UIST, pp. 11–20. ACM Press, Vancouver (2003)
11. Chellapa, R., Jennings, A., Shenoy, N.: A comparative study of mobility prediction in fixed wireless networks and mobile ad hoc networks. In: Proc. ICC, pp. 891–895 (2003)
12. Cheng, C., Jain, R., van den Berg, E.: Location prediction algorithms for mobile wireless systems. CRC Press, Inc. (2003)

13. De Rosa, F., Malizia, A., Mecella, M.: Disconnection prediction in mobile ad hoc networks for supporting cooperative work. IEEE Pervasive Computing 4, 62–70 (2005)
14. Erbas, F., Kyamakya, K., Steuer, J., Jobmann, K.: On the user profiles and the prediction of user movements in wireless networks. In: Proc. PIMRC, pp. 2282–2286. IEEE, Lisbon (2002)
15. Hadjiefthymiades, S., Papayiannis, S., Merakos, L.: Using path prediction to improve TCP performance in wireless/mobile communications. IEEE Communications Magazine 40, 54–61 (2002)
16. Pack, S., Choi, Y.: Fast handoff scheme based on mobility prediction in public wireless LAN systems. IEE Proceedings on Communications 151, 489–495 (2004)
17. Wu, S.-Y., Fan, H.-H.: Activity-Based Proactive Data Management in Mobile Environments. IEEE Transactions on Mobile Computing 9, 390–404 (2010)
18. Chan, J., Seneviratne, A.: A Practical User Mobility Prediction Algorithm for Supporting Adaptive QoS in Wireless Networks. In: Proc. ICON, pp. 104–111. IEEE Computer Society (1999)
19. Das, S.K., Cook, D.J., Battacharya, A., Heierman III, E.O.: The role of prediction algorithms in the MavHome smart home architecture. IEEE Wireless Communications 9, 77–84 (2002)
20. Petzold, J., Bagci, F., Trumler, W., Ungerer, T.: Next Location Prediction Within a Smart Office Building. In: Proc. Pervasive 2005, Munich, Germany (2005)
21. Roy, A., Das, S.K., Basu, K.: A Predictive Framework for Location-Aware Resource Management in Smart Homes. IEEE Transactions on Mobile Computing 6, 1270–1283 (2007)
22. Ziebart, B.D., Maas, A.L., Dey, A.K., Bagnell, J.A.: Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. In: Proc. UbiComp, pp. 322–331. ACM, Seoul (2008)
23. Facebook Really Wants to Know Where You Are, Considers Buying Loopt, `http://www.fastcompany.com/1562563/loopt-facebook-acquisition-rumor-location-based-advertising-social-networking`
24. Location, Location, Location, `http://blog.twitter.com/2009/08/location-location-location.html`
25. Barkuus, L., Dey, A.: Location-Based Services for Mobile Telephony: a Study of Users' Privacy Concerns. In: Proc. INTERACT, pp. 709–712. IOS Press, Zurich (2003)
26. Iachello, G., Smith, I., Consolvo, S., Abowd, G.D., Hughes, J., Howard, J., Potter, F., Scott, J., Sohn, T., Hightower, J., LaMarca, A.: Control, Deception, and Communication: Evaluating the Deployment of a Location-Enhanced Messaging Service. In: Beigl, M., Intille, S.S., Rekimoto, J., Tokuda, H. (eds.) UbiComp 2005. LNCS, vol. 3660, pp. 213–231. Springer, Heidelberg (2005)
27. Voong, M., Beale, R.: Representing location in location-based social awareness systems. In: Proc. BCS-HCI, pp. 139–142. British Computer Society, Liverpool (2008)
28. Dufková, K., Boudec, J.-Y.L., Kencl, L., Bjelica, M.: Predicting User-Cell Association in Cellular Networks from Tracked Data. In: Fuller, R., Koutsoukos, X.D. (eds.) MELT 2009. LNCS, vol. 5801, pp. 19–33. Springer, Heidelberg (2009)
29. Begleiter, R., El-Yaniv, R., Yona, G.: On Prediction Using Variable Order Markov Models. Journal of Artificial Intelligence Research 22, 385–421 (2004)
30. Cleary, J.G., Teahan, W.J., Witten, I.H.: Unbounded length contexts for PPM. In: Proc. Data Compression Conference, Snowbird, UT, pp. 52–61 (1995)

31. LaMarca, A., Chawathe, Y., Consolvo, S., Hightower, J., Smith, I., Scott, J., Sohn, T., Howard, J., Hughes, J., Potter, F., Tabert, J., Powledge, P., Borriello, G., Schilit, B.: Place Lab: Device Positioning Using Radio Beacons in the Wild. In: Gellersen, H.-W., Want, R., Schmidt, A. (eds.) PERVASIVE 2005. LNCS, vol. 3468, pp. 116–133. Springer, Heidelberg (2005)
32. Skyhook Wireless, `http://www.skyhookwireless.com`
33. Krumm, J., Horvitz, E.: LOCADIO: inferring motion and location from Wi-Fi signal strengths. In: Proc. MOBIQUITOUS, pp. 4–13 (2004)
34. Bolliger, P.: Redpin - adaptive, zero-configuration indoor localization through user collaboration. In: Proc. MELT, pp. 55–60. ACM, San Francisco (2008)
35. Yang, G.: Discovering Significant Places from Mobile Phones – A Mass Market Solution. In: Fuller, R., Koutsoukos, X.D. (eds.) MELT 2009. LNCS, vol. 5801, pp. 34–49. Springer, Heidelberg (2009)
36. Ashbrook, D., Starner, T.: Learning significant locations and predicting user movement with GPS. In: Proc. International Symposium on Wearable Computers, pp. 101–108 (2002)
37. Arithmetic Coding + Statistical Modeling = Data Compression, `http://www.dogma.net/markn/articles/arith/part2.html`
38. Burbey, I., Martin, T.L.: Predicting future locations using prediction-by-partial-match. In: Proc. MELT, pp. 1–6. ACM Press, San Francisco (2008)
39. Antifakos, S., Kern, N., Schiele, B., Schwaninger, A.: Towards improving trust in context-aware systems by displaying system confidence. In: Proc. MobileHCI, pp. 9–14. ACM Press, Salzburg (2005)
40. Gopalratnam, K., Cook, D.J.: Online Sequential Prediction via Incremental Parsing: The Active LeZi Algorithm. Intelligent Systems 22, 52–58 (2007)