Context-awareness

Seminar Context-Aware Computing

Sinja Helfenstein, Edoardo Beutler

2006 - Universität Zürich - Prof. Dr. Abraham Bernstein

Contents

1 Abstract

In this seminar paper we discuss two aspects of context awareness. In a first section, we look at the topic from an application perspective: how can context awareness add value to classical applications. We have looked at two examples of recommender systems that include context aware elements - one with rule based recommendations and another one using collaborative filtering for generating predictions. In the second part, we focus on context discovery, sensing and extraction. For this we discuss a research project that aims to improve location recognition for mobile devices that do not have GPS access. The research group showed that it is possible for a mobile device to detect the most important places a user keeps visiting and recognise them as he returns. The motivation for this discussion was to find out whether this approach could be of use in the area of recommender systems. As recommender systems' main purpose is to help the user in places he has never been to before - not in repeating patterns - the intuitive answer would be that it might not add much value to recommender systems. We show that this is not true and that a good recognition of known places can indeed be very useful to increase the performanca and accuracy of mobile recommender systems.

2 Context Aware Systems

The main goal of context aware systems is the provision of information relevant to the user, based on its current situation and context. To get a clear and common understanding of what context actually is, Dey et al. [4] defined it as following:

> Context is any information that can be used to characterise the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a suer and an application, including the user and applications themselves.

While this is a very general definition, Schmidt et al. [8] have specified concrete categories as shown in the following figure 1. According to this scheme every context item is either referring to human factors or the physical environment. Having each three subcategories that can be further split up into unbounded levels of detail allows us to categorise any information relevant to the system within its context space. This is the categorisation that will be used in the following section to evaluate the discussed systems.

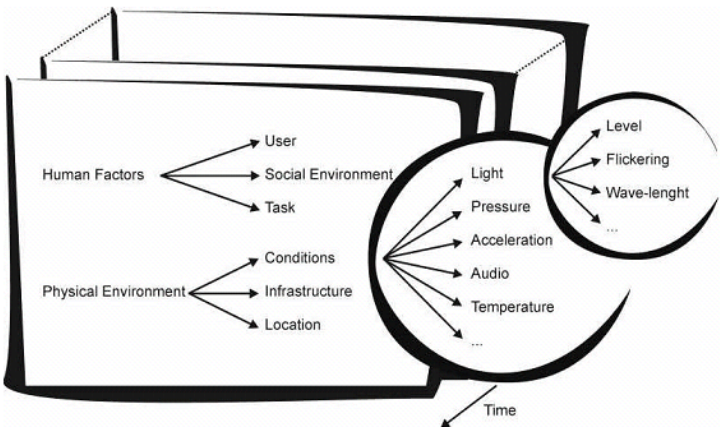


Figure 1: Context Feature Space [8]

3 Context Aware Recommender Systems

The main task of classical recommender systems is to proactively suggest and prioritise items the user may be interested in.[2]. For this purpose information has to be accumulated and filtered based on the user's individual needs. The first challenge is collecting enough information for being able to serve a broad set of users. Based on this information the system has to identify the information that is actually relevant to the current user. Whether this goal is achieved can only be decided by the user himself, as every user has individual expectations and preferences. As the simplified model in figure 2 illustrates, recommender systems always need a set of candidate items and a snapshot of the current context as input. They use their computational model to create a recommendation and return it to the user (output).

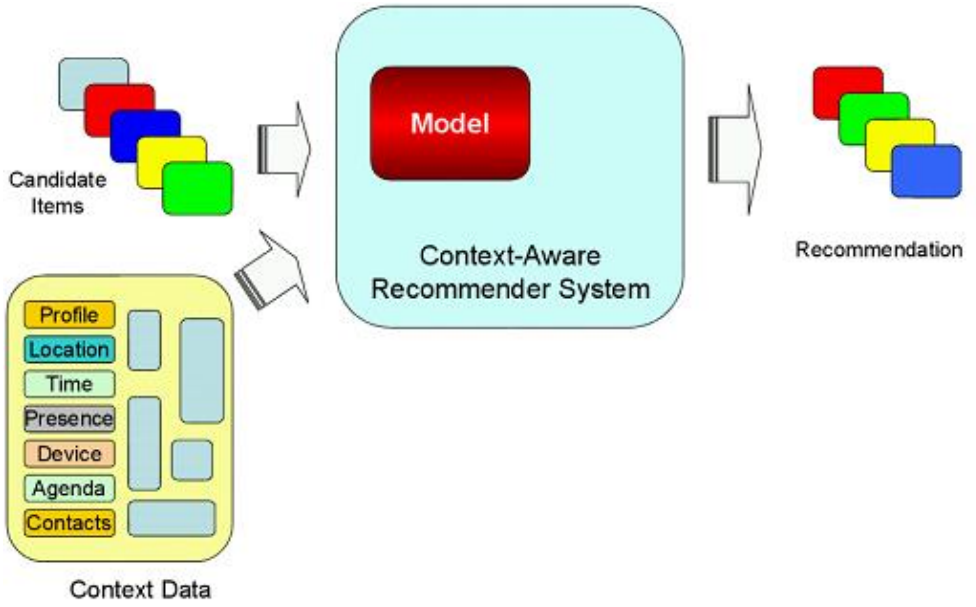


Figure 2: A Context Aware Recommender System [2]

Although the models used for deciding upon the relevance of information items are strongly different, it is possible for all models to divide the criteria used on the context data into

two categories: While **hard criteria** are those that definitely exclude items from further evaluation, **soft criteria** rank the remaining items with respect to their predicted relevance for the user.

The extension of recommender systems with context awareness enriches the amount of input and therewith enables the system not only to rank the information items based on the user's usually static profile, but also on its dynamically changing situation. This should intuitively result in systems that can give diverse recommendations to the same user at two different points in time.[3] Another result of this extension is the integration of the conflicting properties of intelligence and mobility. As the prime goal of a context aware system is to serve people on the move it cannot exceed the size of a pocket (who would want to carry his mobile device in a designated backpack?). By contrast, the main asset of recommender systems is their (artificial) intelligence, which implies more computing power - requiring bigger size, more data, higher power consumption. [10]

One attempt to integrate context awareness with mobile recommender systems has been conducted by Mark van Stetten, Stanislav Pokraev and Johan Koolwaaij at the Telematica instituut in Enschede, Netherlands [9]. With their experiment 'COMPASS' (standing for **Co**ntext Aware **M**obile **P**ersonal **Ass**istant) they built a context-aware mobile tourist application that adapts services based on the users interests and current context. As tourists are by definition most of the time in places they have never been to before they are a good target group for integrating recommender systems with context awareness.

A second approach discussed here is the system UbiMate. Having been built by Annie Chen at the IBM Reasearch Lab in Zurich it is a mobile recommender system that is not using rule-based recommendations but collaborative filtering to generate predictions. [3] Using

this approach can promise easier handling of context information and less maintenance efforts.

3.1 COMPASS

The COMPASS application is running on an open platform called WASP, which controls and coordinates network access, context fetching, service provisioning and recommendation giving. All these tasks are done by stand-alone modules that integrate into the WASP platform. This architecture allows the integration of independent third-party services and the usage of multiple recommender algorithms, as well as the connection of multiple mapping services conforming to the platform standard. The experiment comprises features shown in figure 3. The authors also mention a feature that recognises the devices speed and uses this information to conclude what transport media the user is currently using. According to this information the system could then for example increase the zoom and extend the area regarded as 'local'.

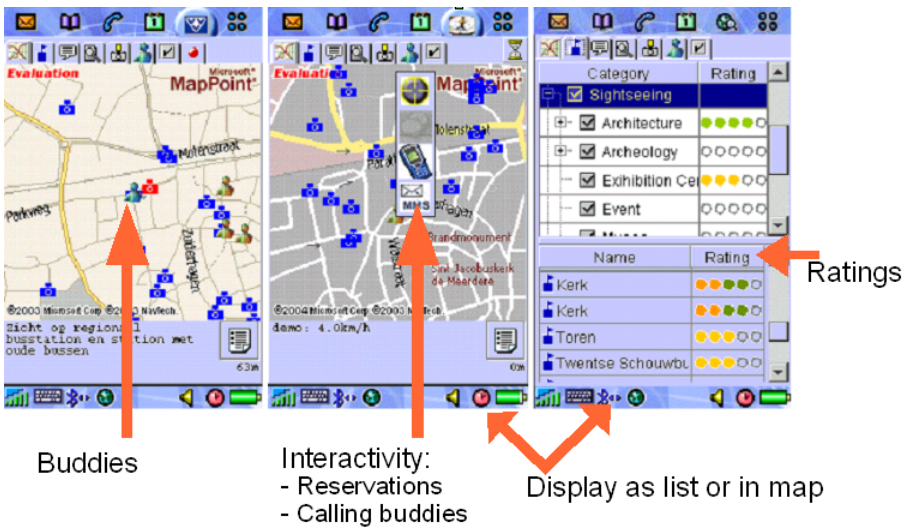


Figure 3: Features of COMPASS

Matching the shown features with the scheme discussed and further categorising the identified context items as either hard or soft criteria leads to the conclusion that the only dynamic information used is the users location, as obtained from the GPS module. As this information is used as hard criteria, the only context information for ranking (soft criteria) is the user's profile. From a recommendation-system point of view this is not significantly different from classical recommendation systems running on stationary computers.

3.2 UbiMate

When trying to include more context information than 'only' location - such as the current weather, the users' social environment, etc. - the system designer faces the challenge of how to collect, model and maintain all this new context information.

3.2.1 Collaborative Filtering

One approach to handle information flood without having to manually collect, enter and define rules of all items is the technique of collaborative filtering used by UbiMate. The underlying method is well known from systems such as amazon.com. Using statistical methods to predict what items a user will most likely prefer, they manage to create predictions without an administrator having to explicitly rate or tag information items with meta information. The users of the community rate the items they already know. Based on their user profile, previous ratings and behaviours the system can then predict how much they will like other items and how much other user's will like the corresponding items respectively.

The process is as following: [3]

1. **Building user profile:** Learn users' preferences implicitly by observing his behavior and explicitly by asking for feedback after every activity
2. **Measuring similarity:** Compare ratings with other users' on the same items to find similar profiles (neighbors)
3. **Generating a prediction:** Predict an item based on weighted sum of like-minded users' ratings for that item

3.2.2 Context used

UbiMate is a system that includes the following context information.

- User Information: Profile and Rating History
- Social Environment: Alone, Friends, Family, etc.
- Tasks: The desired Activity selected
- Location: Coordinates
- Weather: Derived from coordinates
- Time: Time of interaction

Some values must be entered manually by the user (social environment, tasks), while others are being provided a standard value from either the system (time), the user's profile

(location) or inferred from other context data, using related content providers (weather inferred from location).

The new aspect introduced by context awareness is that the collaborative filtering algorithm must not only consider the user's habits, but also a snapshot of the current context along with the rating. If user A gives a good rating for item X at bad weather, the system should not recommend this specific item to user B at rainy weather, even if in other respects users A and B are identical in what they like. The same holds for a restaurant that is great to visit with fellow students, but less appropriate with grand-parents.

The more dimensions included in the prediction generation, the better the predictions get. To allow this scalability a snapshot of context is represented as composite of different context data, which may be available or unavailable and must therefore be stored independently.

The statistical means used in most collaborative filtering systems is the person correlation coefficient that measures the similarity of two users. The more similar they are, the more the recommendations they receive will depend on each others previous ratings. The same is done with the different context values - taking the Pearson Correlation Coefficient tells the system how strongly a pair of context types depend on each other.

Further details on context modeling and comparison are not discussed here, as they can be found in the original papers by A. Chen. [3].

3.3 Evaluation

The following figure 4 uses the categorisation by Schmidt et al. [8] (as previously introduced in section 2) to evaluate what types of context have been used in the discussed systems. In addition to this categorisation the items are divided into hard (H) and soft (S) factors.

As figure 4 shows, location is the only context the two systems share in the category of physical environment. All the further context information that is used is either derived from or used to induce to the users location (weather, city area, speed). In addition, UbiMate includes the factor time to evaluate the activity. All the other factors applied are the user specific human factors. In all cases they need to be entered manually by the user, meaning that they are not acquired autonomously by the system. Consequently, there is much room for improvement in the area of context recognition. One example from this research area will be discussed in section 4

		COMPASS		UbiMate	
		Hard	Soft	Hard	Soft
Human Factors	User		Explicit interests		Implicit profile based on previous ratings
	Social Environment				Company
	Tasks	Subscriptions		Activities (Subscriptions)	
Physical Environment	Location	Coordinates			Coordinates City Area
	Infrastructure				
	Physical Conditions	(Speed)			Weather Time

Figure 4: Evaluation of the two systems

3.4 Conclusion

Having studied the two discussed systems we conclude that even very recent research projects are not yet very sophisticated when it comes to automated context fetching. In return, there are very advanced methods of dealing with the once-acquired context information through flexible, reusable and extensible models for context data. Especially the application of collaborative filtering seems very promising for handling the multi-faceted context data without having to initially define semantics for each item.

Regarding the economical success of such systems we see two main problems that should be addressed: As already mentioned the context fetching is still in the fledgling stages which requires the user to enter much information manually. This affects the user experience and diminishes the his benefit from the system. The second challenge would be achieving a critical mass of participants in order to get a big-enough set of data to generate good predictions.

As the second problem is more of an economical background and does not much differ from the challenges faced by classical recommender systems, we will not discuss it any further. Instead we will look closer at one project that tries to improve the automatic context recognition.

4 BeaconPrint

4.1 Introduction

The examples of context aware systems showed one important issue to deal with: The geographic location of the user (to be exact the location of the device, the context aware application is running on), recognizing when a user stays in a place or returns often to it and, as well, assigning a colloquial name to this place (”Home”, ”University”, ...), instead of ”non-talking” coordinates. In this section a location method based on WiFi and GSM will be presented. The application recognizes also the important places in the users life. There will be no discussion of the third task mentioned, the name assignment problem (automatically assigning names, geocoding). To fulfill this tasks two algorithms will be needed.

4.2 Place Learning and Place Recognition Algorithms

Place learning algorithms take waypoints (location data collected from sensors) as input. Such an algorithm has two tasks to do:

1. **Define waypoints** by segmenting the sensor input into times when a device was in a stable place (in comparison to a mobile state e.g. when the user is on a train).

2. **Merge multiple waypoints**, captured in one place at different times.

The recognition algorithms takes the waypoints to consider whether the device returns to a known place.

1. **Recognize** if the device returns to a known waypoint.

2. **Recognize** a mobile device. A device which is not in a stable place.

The sensors captures location data. This data goes to both, the place learning and the recognition algorithm. The place learning algorithm builds with the data a waypoint list which is the second input source of the place recognition algorithm.

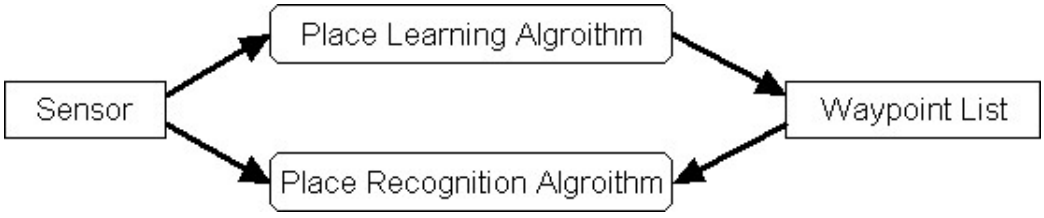


Figure 5: How the place learning and the place recognition algorithms work together.

4.2.1 BeaconPrint

The BeaconPrint place learning and recognition algorithm [5] uses, in contrast to the most others, not GPS signals to locate the device, but GSM and 802.11. This allows a more accurate location in the ”skyscraper canyons” of cities where (often) no GPS signals are available.

4.3 Previous Approaches

Here some earlier approaches for place learning and recognition. The accuracy of Beacon-Print was tested comparing it to these algorithms.

4.3.1 GPS Dropout plus Hierarchical Clustering Algorithm (Ashbrook and Starner)

Ashbrook and Starner's algorithm [1] is exploiting the already mentioned fact that GPS is often unavailable indoors or in cities. There are two conditions under which the algorithm recognizes a position as stable:

1. The GPS signal is lost for a certain time (given as a parameter t to the algorithm).
2. The speed of the device stays below one mile per hour.

In both cases it is assumed that the user has stopped (maybe walks only from his desk to the printer once in a while) or is inside a building where there is no GPS signal. When none of the two conditions is fulfilled, the device is in a mobile and not in a stable place. One of the problems of this method is, that it might define wrong places as significant. To show such a situation here an example: Let us suppose I work, as a waiter in a bar, in Zürich main station. On my way to work, the train enters the tunnel near Stettbach from where it stays underground. That is probably the point where I would loose the GPS signal. The signal will be lost until evening on my way home, when the train leaves the tunnel at the same position. In such a case A&S's algorithm will recognize a significant place not at the main station, where I work, but at the tunnel entry near Stettbach.

4.3.2 comMotion Recurring GPS Dropout Algorithm

In contrast to the two other examples which are only algorithms, comMotion is an actual device (wearable or hand-held) [7]. It is a so called remembrance agent and has integrated a

GPS receiver. The comMotion device can present information, relevant to a user, depending on the place he is (e.g. a list of the courses starting soon when he enters the university). As Ashbrook and Starner's algorithm also comMotion works with signal losses, but in a different way. The comMotion device recognizes a place as stable if the GPS signal is lost at lest three times within a certain radius. The device is mobile if no drop signals can be found.

4.3.3 Sensor-Agnostic Temporal Point Clustering Algorithm (Kang et al.)

While the two other algorithms exploited the properties of the GPS signal, this one [6] tries to avoid this dependence. It uses a time- and distance-based clustering to extract the important points out of the location data. The technology independence of this algorithm is also its weakness. When using GPS data as input, it has problems to deal with urban environments, due to the lossy signal, the fact, the other algorithms exploit. Depending on capture time and distance the data is grouped into clusters of significant points. When no cluster can be found in a certain distance, the state is recognized as mobile.

4.4 How BeaconPrint works

As already mentioned BeaconPrint uses the 802.11 (WiFi) or GMS radio signals for the location. There are two advantages in using these technologies:

1. WiFi or GSM are more common than GPS. There are already many mobile devices with one of them built in.

2. The signals of WiFi and GSM are more available especially in urban regions and / or in buildings.

The localisation is done by scanning the broadcast of identifier signals from the WiFi access points and GSM towers. These fixed sources of radio signals are called beacons. The signals the device receives build the input for the learning algorithm. The BeaconPrint algorithm checks the input data. If the device stays for a time window of at least w in the same place, this place is marked as significant and a fingerprint of is generated. The fingerprint is a list of all timestamped beacon IDs received at the stable place. When the device returns to a place and another fingerprint is generated, the algorithm will merge them.

One problem of this technique is that sometimes beacons are recognised as new although they are already known. These can happen with beacons with low response rate, which are only seen every few scans. They can not be separated from new beacons received while leaving the place. This leads to errors in the recognition or in splitting a place into several places. This effect can be reduced by using a certainty parameter $c \in [0 \dots max]$. The window is divided into so called dwell increments ($d = w/c_{max}$). If such a dwell increment passes without receiving a new beacon, the scan is stable and a (not very certain) fingerprint is collected. The certainty of the fingerprint increases every subsequent dwell passing without seeing a new beacon (beacon that is not in the current fingerprint and has not been seen for at least w). If a new beacon is seen during a dwell the certainty is decreased. The fingerprint gets valid, when c reaches c_{max}. If c is decreased until 0 the fingerprint is recorded if it is already valid, if not it will be discarded. When $c = 0$ BeaconPrint removes beacons with less than c_{max} entries from the fingerprint. This is done to get rid of the e.g.

new beacons which ended the fingerprint when leaving the significant place.

As mentioned above, there are two parameters the BeaconPrint algorithm uses:

1. **time window** w. w indicates the minimum time the device has to stay in a place to make it a significant place. Choosing w to low (e.g. one minute) will result in many insignificant places, recognised as significant ones (e.g. the bus station, traffic lights, the spot you had to retie your shoe, ...). Is w to high some significant places will be left out (e.g. drinking a tea every morning your favourite tea room).

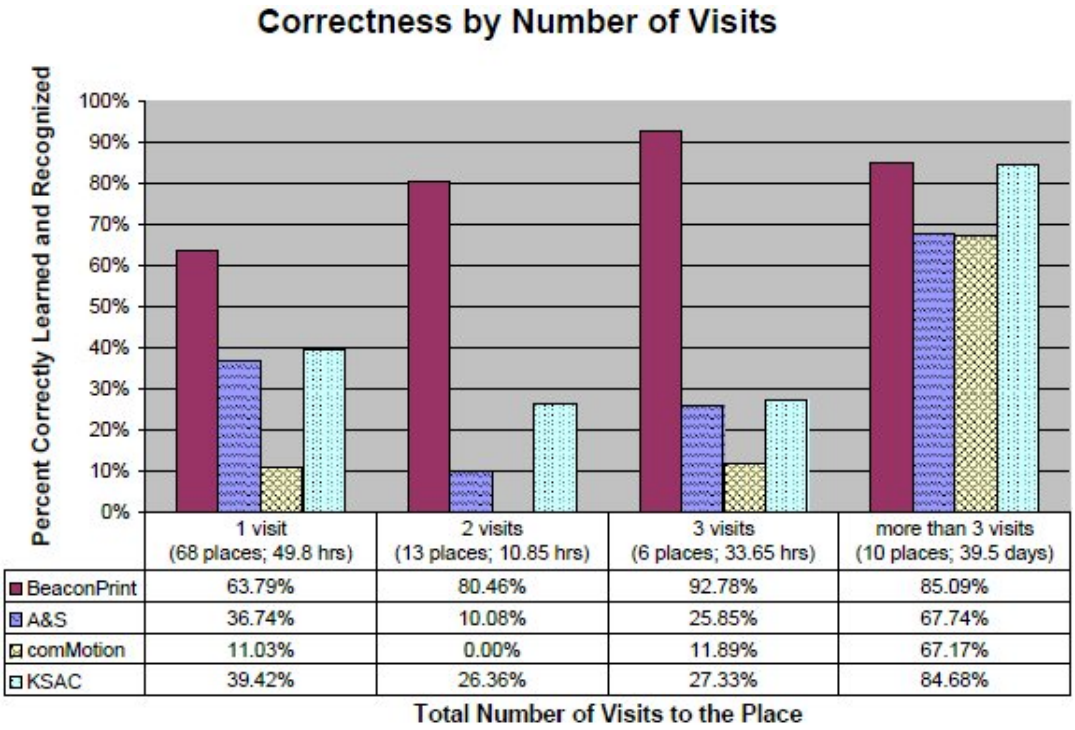
2. **confidence depth** c_{max}. If C_{max} is chosen to low, infrequent beacons can lead to fragmented places where there should be one (to solve this problem this parameter was introduced). Choosing a to high value for c_{max} results in merged places (e.g. between two lectures you have lunch in a restaurant near the University. BeaconPrint will merge the places and not be able to distinguish between them).

4.5 Evaluation of BeaconPrint

BeaconPrint has been tested and compared with the three algorithms mentioned above in 4.3.

4.5.1 Evaluating the Algorithm

To evaluate the algorithm three members of the research team walked around with a backpack containing a GPS, a WiFi and a GSM devices. For a month these devices collected their location data 24x7. In addition the data collectors also made a diary of the



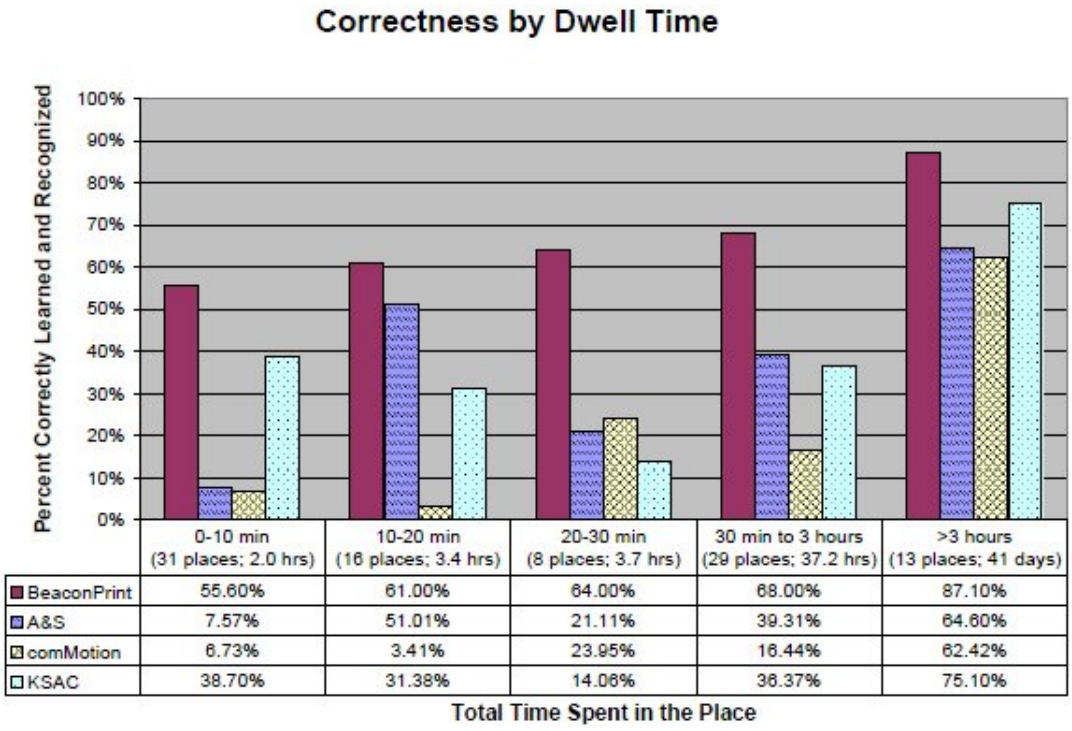
	1 visit (68 places; 49.8 hrs)	2 visits (13 places; 10.85 hrs)	3 visits (6 places; 33.65 hrs)	more than 3 visits (10 places; 39.5 days)
BeaconPrint	63.79%	80.46%	92.78%	85.09%
A&S	36.74%	10.08%	25.85%	67.74%
comMotion	11.03%	0.00%	11.89%	67.17%
KSAC	39.42%	26.36%	27.33%	84.68%

Figure 6: Performance by number of visits. Taken from [5]

places they visited during this month. Afterwards the four algorithms were fed with the data and their results were compared with the data from the diaries. The data collectors can be grouped into three categories: first the ”family guy” Adam, second is the ”urban single” Bob (most of his significant places are clustered in a small area) and third Charles, the ”traveller”. The first half of the collected data has been used to learn the places. The second half was used to check whether the places where recognised correctly.

Bob’s data posed the most problems to all four algorithms. His urban environment with lots of beacons on a small space combined with unpredictable radio propagation between the steel canyons of the buildings made the location a hard task. Nevertheless BeaconPrint performed best of the tested algorithms. As we can see in figure 6 BeaconPrint performed especially good (compared to the others) in recognising places the device was only few times (less than four times) while also the other algorithms had a good recognition rate

on the common places like ”home” or ”work”. In figure 7 we see that also for places with a short dwell time BeaconPrint succeeds a lot more often than the other algorithms. In



	0-10 min (31 places; 2.0 hrs)	10-20 min (16 places; 3.4 hrs)	20-30 min (8 places; 3.7 hrs)	30 min to 3 hours (29 places; 37.2 hrs)	>3 hours (13 places; 41 days)
BeaconPrint	55.60%	61.00%	64.00%	68.00%	87.10%
A&S	7.57%	51.01%	21.11%	39.31%	64.60%
comMotion	6.73%	3.41%	23.95%	16.44%	62.42%
KSAC	38.70%	31.38%	14.06%	36.37%	75.10%

Figure 7: Performance by time spennt at a place. Taken from [5]

conclusion we can say that BeaconPrint’s mayor step forward, in comparison to the other algorithms, is the ability to recognise significant places after only few visits and also were only visited for a short time (e.g. your doctor, your car wash, ...).

4.5.2 Evaluation Completion Survey

Knowing that a test with three participants is not really representative, a survey study about the places people go was accomplished. The three data collectors plus six external participants were interrogated about the places they go at least two times a year in a radius of 50 miles from their home. The results of the survey in figure 8 shows that, especially for the frequently visited places the data collectors seem to be representative. In seldom

visited places there is a gap between the data collectors and the survey participants which may be based on the data collector's "higher sensibilisation" for the topic.

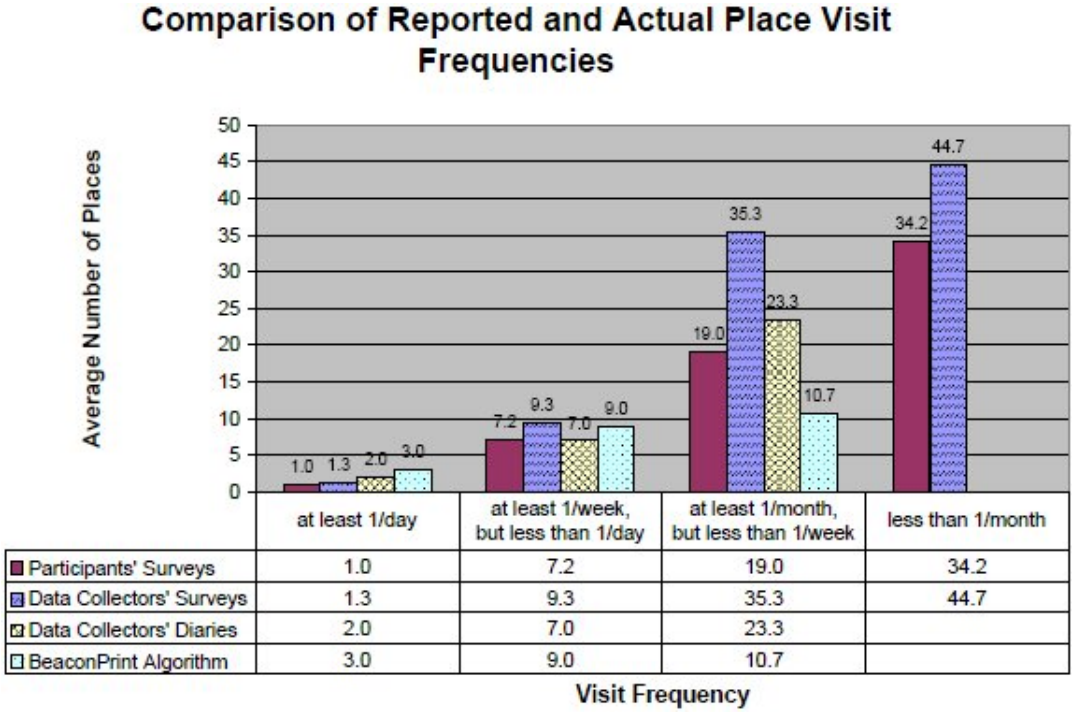


Figure 8: [5]

4.6 BeaconPrint and Recommender Systems

At the first glance BeaconPrint seems to be more or less useless for recommender systems. The recommender systems are basically thought for people new to a place, and not for those who already know a place and return to it. Normally the "returners" do not need recommendations. In that view one important point has been missed.

BeaconPrint is not limited to one device. If the collected fingerprints of a location are coupled with the recommendations a user makes, the device of a user, visiting a place for the first time, can recognise this place as known and offer the appropriate recommendations. Although the user has never been there, the device recognises the place, learned by other

users.

BeaconPrint seems to be a good choice for such systems for its good performance on rarely visited places. It is enough if only two to three people recommend a place to recognise it with a high accuracy. Lowering the amount of data needed to make useful recommendations could be a huge step towards the success of a system. One of the mayor problems of such systems is, to reach a critical mass of users to make the system useful. Thus it gives a system an advantage if it needs fewer users to make good recommendations.

5 Conclusion

After a short introduction to context aware systems we took a closer look on two examples of recommender systems. We concluded that even recent systems need the input from the user and are only able to collect parts of the context by themselves. We also noticed that, once the data is collected, there are many sophisticated solutions to handle and use the collected information. We filtered out two main issues recommender systems have to deal with, first the mentioned lack in automated context recognition, diminishing the user's benefit and second the critical mass of participants to make the system a success.

In the second part we presented BeaconPrint, an algorithm to learn and recognise significant places in a user's life. Unlike many other algorithms BeaconPrint is WiFi and GSM based, thus making it a good choice for urban environments where GPS is often unavailable. We concluded that BeaconPrint, by enhancing the context fetching and the recognition of known places could have a positive influence on the second problem mentioned about recommender systems. We think that, if the collected data is more precise

and a device returning to this place can be predicted more accurate, this will decrease the critical mass of users needed to get enough data to generate good predictions.

6 Appendix

6.1 Authors and Responsibilities

Edoardo: LaTex Template, Beacon Print, Conclusion

Sinja: LaTex Template, Introduction, Recommender Systems

References

[1] D. Ashbrook, T. Starner: Using GPS to learn significant locations and predict movement across multiple users. Personal and Ubiquitous Computing 7 (2003) 275-286

[2] L. Buriano, M. Marchetti: The Role of Ontologies in Context-Aware Recommender Systems

[3] A. Chen: Context-Aware Collaborative Filtering System: Predicting the Userś Preferences in Ubiquitous Computing

[4] A. K. Dey, G. D. Abowd: Towards a Better Understanding of Context and Context-Awareness

[5] J. Hightower, S. Consolvo, A. LaMarca, I. Smith, J. Hughes: Learning and Recognizing the Places We Go

[6] J. H. Kang, W. Welbourne, B. Stewart, G. Borriello: Extracting places from traces of locations. In: Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH 2004), Philadelphia, PA, ACM Press (2004) 110-118

[7] N. Marmasse, C. Schmandt: Location-aware information delivery with commotion. In: Proceedings of the Second International Symposium on Handheld and Ubiquitous Computing. Volume 1927., Springer-Verlag (2000) 151-171

[8] A. Schmidt, M. Beigl, H-W. Gellersen - University of Karlsruhe, Germany: There is more to context than location.

[9] M. Van Stetten, S. Pokraev, J. Koolwaaji - Telematica Instituut, The Netherlands: Context-Aware Recommendations in the Mobile Tourist Application COMPASS.

[10] A. Zaslavsky - Monash University, Australia: Mobile Agents: Can Tey Assist with Context Awareness? In: Proceedings of the 2004 IEEE International Conference on Mobile Data Management