Enabling Proactiveness through Context Prediction

Petteri Nurmi[1], Miquel Martin[2], John A. Flanagan[3]

[1] Helsinki Institute for Information Technology HIIT
Department of Computer Science, P.O. Box 68
University of Helsinki, FIN-00014 Finland
ptnurmi@cs.helsinki.fi
[2] NEC Europe Ltd.
Kurfuerstenanlage 36, 69115 Heidelberg, Germany
miquel.martin@netlab.nec.de
[3] Nokia Research Center
P.O. Box 407, FIN-00045, NOKIA GROUP, Finland
adrian.flanagan@nokia.com

Abstract. In order to enable proactiveness in applications and services, inferences about likely future contexts are needed. This task, which is often referred to as context prediction, has not been properly addressed in the literature. We define the different components of context awareness from data acquisition to prediction. Some mathematical principles of making predictions, alongside with technical challenges that must be solved in systems producing predictions about context are discussed. Several possible uses for predicted context are presented as well as some architecture issues related to the implementation of context prediction.

1 Introduction

Context awareness is starting to play an increasingly important role in modern software systems, especially in software for wireless information devices. Context, according to Dey and Abowd, is any information that can be used to characterize the situation of an entity, where an entity is a person, place or object that is considered relevant to the interaction between a user and an application [1]. Together with the ubiquitous computing paradigm, which strives for making computing systems available anytime and anywhere [2], context-awareness offers a powerful vision that aims at building novel applications and services that adapt according to the situation of the user.

In order to further enhance this vision, ways to modify typical interaction patterns between humans and applications are needed. One of the most promising contemporary approaches is *proactiveness*, which attempts to reduce required user efforts by recognizing external stimuli and reacting automatically to the relevant ones [3]. Enabling proactiveness requires information about the users' future needs and thus inferences about the users' future contexts are needed. Within the field of time series analysis (e.g. [4, 5]), this task is often referred to as forecasting or prediction. Although a distinction is sometimes made in the literature both terms are interchanged throughout the paper.

When the used data contains only real valued measurements, a wide class of methods exists for making predictions. However, in ubiquitous settings the data arrives from rather heterogeneous sources and thus it is sometimes hard to map the data into meaningful real valued representations. For example, if MAC addresses or service descriptions are mapped into real values, distances between points become meaningless and standard prediction methods are not applicable anymore. Additional challenges arise from the limited capabilities of the devices that calculate the predictions. Namely, mobile and handheld devices have limited resources such as power, memory and processing power. In addition, due to mobility, network connectivity is often limited.

To the best of our knowledge, the only approach thus far that addresses the above mentioned domain specific challenges is by Mayrhofer et al. [6, 7, 8], who introduce a layered architecture to make predictions about the entire context of a user. The first layer of the architecture is responsible for gathering sensor data and mapping sensor readings into feature values such as the mean or variance. Next a classifier is used to assign class membership values for different context classes. Learning the context classes is the task of a suitable classification algorithm. The class membership values are then fed into the prediction layer which infers likely future values for the class membership values. Finally, both the original and the predicted class membership values are given to the labelling layer which assigns semantic meanings to the values in a semi-supervised manner.

Also some work has been done on applying prediction methods for contextual data. However, the predicted contexts are typically location-related and restricted to a specific environment (smart space or grid). For example, Laasonen et al. [9] define a hierarchy of locations and describe various methods using statistics for predicting next locations. Patterson et al. [10] use a dynamic Bayesian network for predicting likely travel destinations on a city map. However, the algorithm requires the city map and is thus restricted to a particular area. Mozer et al. use neural networks to predict, e.g., how long a user stays home and whether a particular zone becomes occupied [11]. Markovian models have been used by Kaowthumrong et al. to predict what remote control interface the user is likely to use next [12]. The predicted interface was then automatically loaded to the device. Petzold et al. [13] use global and local state predictors for predicting the next room the user is likely to enter in an office environment. A more extensive methodological comparison was done by Mayrhofer [8], who compared the performance of different methods such as neural networks, Markov models, ARMA forecasting and support vector regression. However, the tests were performed on a specific data set and thus the results cannot be fully generalized.

The role of predicted information in context-aware systems is often merely seen as an enabler for automatic service and application execution or preparation. In this paper we identify uses of context prediction and discuss the technological and other challenges in using the predicted information. The goal of this paper is to give a thorough treatment of issues related to context prediction. We begin by discussing mathematical and technical aspects of contextual prediction in Section 2. In Section 3 we outline different possible uses for predicted contextual information, whereas Section 4 is reserved for discussion about additional issues, such as sharing predictions and platform needs. Finally, Section 5 concludes the paper and discusses future work.

2 The Context Prediction Process

As illustrated in Fig. 1, context prediction is a multi-stage process, where individual stages correspond to a particular data mining / data analysis task [14]. Enabling prediction capabilities in context-aware systems requires providing support for all of the individual stages. In this section we discuss the individual stages on a generic level and identify technical requirements and challenges arising from the tasks. A technical discussion is undertaken in Section 4.1.

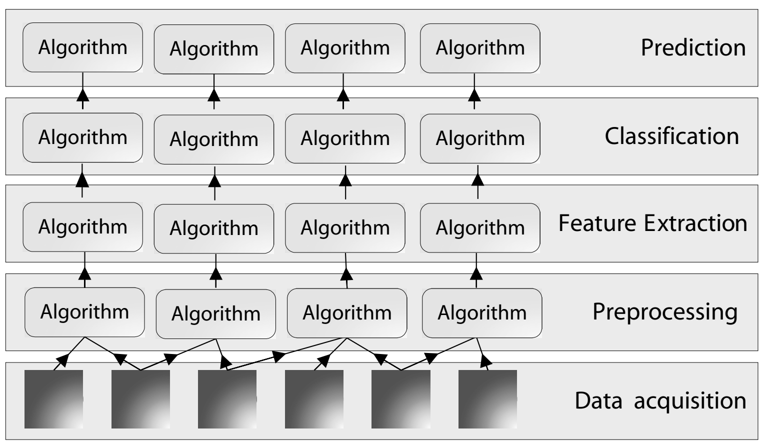

Fig. 1. The context prediction process.

Data acquisition The first step in any data analysis task is naturally gathering data. In context-aware systems this task is however a bit more complicated as the different sources of context data can be very heterogenous, they can be scattered around the environment and access to the data might require using sensor specific interfaces. This scattering of sources raises issues related to privacy and security as unauthorized access to data must be prohibited. Also the user might want to anonymize data, when network connectivity is unsecure or when they do not fully trust the device(s) performing the next steps of the processing.

Existing solutions for data acquisition are rather diverse and they have been designed for different settings. For example, the framework proposed by Mayrhofer [8] does not have a separate data gathering layer, but common data access interface is provided only after the preprocessing stage. In addition, because all the processing has been designed to take place on the devices, security and privacy aspects are not pervasive by nature. Thus, this architecture is not applicable in our setting, where sensing devices do not necessarily reside on the terminal device. On the other hand, the *context toolkit* [15] offers a solution where so-called *widgets* are used for encapsulating data acquisition from individual sensors. However, this framework does not encapsulate support for privacy and security and, in addition, the framework suffers from severe communication overhead.

Preprocessing After the data has been gathered, preprocessing algorithms are applied to the data (e.g. outlier removal, normalization) [16]. The actual algorithms that are applicable for a specific task are dependent on the nature of data and, to this end, Mayrhofer [8] suggested a framework, where the preprocessing is integrated with data gathering so that the selection of algorithms is domain-based. Other existing work mainly uses a blackbox approach, where the processing tasks are integrated to a common component and thus the used algorithms cannot be easily reconfigured at runtime. As state of the art implementations of feature extraction and classification stages are similar to the implementations of the preprocessing stage, further discussion about existing work is omitted in these subsections.

Feature Extraction After preprocessing techniques have been applied to the data, the next step is to derive features from the measurements using statistical and signal processing techniques. The algorithms that are used in this phase are typically aggregating by nature, i.e., they produce a single value, such as the mean or variance, from a set of measurements. In order to reduce communication and computational overhead, the amount of measurements needs to be reduced. To this end, aggregate features i.e. those that calculate a single output from a set of input values, are preferred. However, if the sensor measurements are produced by multiple sources, also features that attempt to capture some aspects of similarity between the sources can be derived.

Classification The next phase of the analysis consists of classifying the features extracted from the sources to determine a particular context class. For example the "walking" activity could be one class determined from an acceleration source. The classification stage can also be used to fuse different context features together to generate higher level contexts, for example by combining activity and location features to determine the "walking downtown" context. Labelling of the classes is an important step and defines the type of output given by the classifier. Furthermore it very much defines the type of algorithms that can be used in the next prediction phase. In the ontology approach to context awareness the desired result from the classifier would be a human understandable label such as the examples used already, "walking" etc. Another option is to label the classes with arbitrary distinct symbols as in [17]. Finally, instead of labelling, each class can rather be represented by an input to the classifier that in some sense is "typical" of that class and this typical input can then be used in the next prediction phase. Finally the classifier can essentially just transfer the input directly to the output which can be understood as each input representing a distinct class.

Prediction Finally, prediction algorithms are applied to the classified context data. As it is unlikely that a common representation can be used for all data, i.e. some values are represented using strings and others using vectors, it is of utmost importance that the prediction phase allows different algorithms to be used based on the nature of data. The interpretation or representation of the context is also influenced by the nature of the prediction algorithm. For example, when the data is vector valued, Kalman filtering (see e.g. [18]) or particle filtering techniques (see e.g. [19]) can be directly used and the output is also continuous. In the case of Hidden Markov Models (HMMs) [20] the input can be either real valued or a set of discrete states and the output is a discrete state. In the case of a discrete state input represented by a symbol string the output is also a discrete state as in [21].

3 Applications of Context Prediction

In typical ubiquitous computing scenarios predicted context is mainly used to anticipate which applications or services the user is likely to use in the near future and launch them in advance. However lots of critique towards this kind of process has been given as this kind of systems easily distract the user; thus offering a worse user experience than a system that does not use the predicted contextual information (see [22, Chapter 3] for a more thorough discussion). However, the predicted information can be used to aid various other tasks, but to date the alternative uses have not been thoroughly addressed. Some alternative uses have been proposed, for example Mayrhofer [8] discusses the possibility to use predicted information in automatic reconfiguration, accident prevention, alerting and planning aid. However, to our best knowledge the use of predicted information in group related applications has not been discussed earlier.

Reconfiguration Although automatically launching applications can easily disturb the user, predicted information can still be effectively used for reconfiguration purposes. For example, system libraries can be automatically loaded and unloaded based on predicted user needs. The usefulness of this comes from the reduced application / service launch times. However, this approach has also severe drawbacks as the communication overhead increases. Thus a suitable combination between the (un)loading and communications must be found.

Another reconfiguration related use of predicted information comes from prioritization of services, actions etc. For example, in order to select the so-called operational context, causal models are used to select potential control actions for the current context. However, by looking further into the future, the proposed control actions can be better prioritized.

Device power management As mobile devices become progressively richer in applications and features and their processing power increases a major bottleneck is the capacity of the batteries to power the devices. Context prediction can serve as a means to distribute the power needs of the device over time in a manner that does not decrease the user experience. For example if a user attempts to send a large MMS in a region of bad radio reception which would require large amounts of power, the context prediction could delay the transmission of the MMS knowing that the user will shortly pass into another context where the radio environment is more favorable.

Early warning In principle, predictions about the future can be used to check whether an application is likely to enter an illegal state. Similarly, it is possible to issue, e.g., traffic alerts etc. based on predictions of the user's location.

Planning aid One of the most important uses of prediction is that of planning aid. Namely, if future needs of the user can be reliably deduced, this information can be used to plan suitable sequences of actions that lead to the desired goal. This information can be either directly shown to the user who then performs the planning himself or the information can be feed to autonomous agents that perform some form of planning on this information. This is particularly useful in the case of actions that need time consuming preparation: proactive service recommendation would give easy access to the service, but prediction of the need might actually save big amounts of time.

Early coordination of individuals Related to the previous subsection, planning can be also done in the context of groups. Namely, if the needs and/or actions of the members of the group can be predicted, the suggested actions can be selected so as to coordinate individual members towards a solution that best meets the interests of the group as a whole.

Aiding resource management Related to the previous two points, if we consider the group to be a set of devices, the coordination can instead refer to aiding resource management for individual users.

Inferences on other's future contexts Finally, a completely novel area which, to our knowledge, has not been discussed earlier, is to use predictions about others future contexts to help current inferences. Namely, the needs of a user might be dependent on the future contexts of others. For example, if a user is using a wall screen to display confidential information, this information needs to be hidden if other users are in the room. Thus, predictions of other people's locations can be used to suggest hiding the information content.

4 Discussion

Prediction of a dynamically changing variable, in any sense, requires a certain level of computational and memory resources. These requirements in a mobile environment place limitations on the complexity of the prediction. In section 4.1 some architectural perspectives related to distributing the computational load associated with context prediction are discussed. Context prediction for a single user is challenging and useful but even more so if the prediction is expanded to include a group of users. Related issues, apart from the increased complexity of the problem, such as privacy and trust are discussed in section 4.2. Perfect context interpretation in all contexts is not possible, and context prediction is even less so. The level of confidence in a prediction is important in deciding how to react to the prediction. Related to this, some measures of prediction confidence are discussed in section 4.3. Finally an agent technology interpretation of context prediction is discussed in section 4.4

4.1 Architecture considerations

Context monitoring and reasoning are resource consuming. Adding prediction and its monitoring certainly requires further resources. The limitations of today's mobile devices, and specially their autonomy, makes it unfeasible to delegate all responsibilities to it, and so, server side processing is required. On the other hand, the amount of variables to monitor and the potential number of users, would present serious scalability concerns on any centralized architecture.

A possible solution would be a hybrid server and peer-to-peer approach, in which a reduced number of users (perhaps in a family or building) entrusts their processing requirements to a shared server, and this network of context servers might interact with each other in a p2p fashion. Keeping geographically close users in nearby servers would promote easier sharing between the context of the people that are most likely to interact with each other.

On the other hand, we are likely to see the flourishing of a new sort of service providers, which specialize in managing context information just like we use a mail server today. Such context network would provide context sharing both between users and within the user devices, and enable group interactions between trusted peers.

Finally, it should be noted that certain applications of context prediction might not require a distributed system, either because of their simplicity or restricted scope. Therefore, the model should be flexible enough to allow for this double approach, depending on the needs and restrictions of the situation.

4.2 Prediction sharing

The focus in context-awareness is now based in the axiom that a user context dictates, to a large extent, what his desires and necessities at any given moment are. Although essentially right, this overlooks the influence of other peers in our own context, and how their restrictions can influence ours. The following example illustrate this effect:

Bob is borrowing Alice's car for the month's shopping later at noon. Unfortunately, Alice's kid returning from summer camp predicts his arrival much sooner than expected; This is shared with Alice who will then predict the need for the car, and informs Bob, who decides to go shopping earlier.

This essentially shows how our context interpretation can benefit from receiving input on the predictions of the rest of parties we interact with. The improvement stems from two sources: first, the fact that each individual has full access to the information related to himself, but is restricted when it comes to others. While Alice might not want to share the details of her life, she is probably happy to give fair warning that her car will not be available. Second, even in the case of having full access, ones own reasoning engine might not be prepared for the circumstances, such as the telltales of a coming storm when in a new country. Any of the locals will have the same data, but a better model to reason on it, and so the prediction could be shared freely.

These two aspects already show the main problems derived from knowledge sharing: the protection of the sender's privacy, and the trustworthiness of the information on the receiver's side. The challenge here resides in unobstrusively distinguishing what can be disclosed or trusted and what is better rejected, without cumbersome monitoring by the user.

This semantic approach at the shared information forces peers to work beyond the symbol processing plane, but does so only on a front filter fashion. A number of solutions have been proposed to solve the problems of privacy assessment and peer trust, as in [23] and [24] and these will not be discussed here.

4.3 Confidence on predicted context

The further into the future we attempt to foresee, the more difficult it usually is. From technical perspective the "horizon", i.e. the point until which we need to predict contextual values, is determined by the system delay. For example, if launching an application takes one minute, then enabling proactiveness would require predictions of context values approximately two minutes ahead (including system delay in the prediction etc.). Furthermore, the more contextual variables are used in the predictions, the more likely it is that the underlying dynamics that govern the changes in the context are chaotic and thus highly unpredictable. Thus for applications it would be necessary to be able to estimate the confidence on the predictions.

In general, confidence is easier to evaluate using subjective methods. However, although the method itself has a natural way of giving confidence estimates, the estimates are often hard to evaluate. Forecasting methods, on the other hand, are usually lacking a "probabilistic connection" and thus the confidence can not be evaluated, but alternative ways are needed. One possibility is to gather some data and to calculate the so-called *empirical Lyapunov exponents* from this data (see e.g. [25]). The Lyapunov exponents come from the theory of dynamical systems and they are used to estimate whether a dynamical system is chaotic or not. If the system is not chaotic, the forecasting methods typically yield reasonable results. However, if the system is chaotic, then the confidence on the estimates decreases exponentially in time.

On the testing stage, it would be preferable to model a framework that is as algorithm independent as possible, so that these could be exchanged modularly as required.

4.4 Agent technologies and context prediction

According to Russel and Norvig [26, p.33] anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators is considered to be an (intelligent) agent. Furthermore, as [3] discusses, these functionalities are also required for enabling proactiveness, and thus agent technologies seem to offer a suitable paradigm for context prediction.

To this end, we define a *context prediction agent* to be an entity that monitors the context of the user, predicts future context values and *autonomously* decides on actions to perform based on the predicted context. Moreover, when contexts of different users are monitored by distinct agents, multi-agent technologies can enable the easy sharing of predicted information and group-based proactiveness.

5 Conclusions and Future Directions

In this paper we have discussed context prediction as a generic problem. First we discussed the mathematical aspects of making predictions, after which an architectural solution towards providing predictions was suggested. Next we discussed various usages of predicted information and finally we discussed some additional issues which affect practical solutions.

Acknowledgements

This work has been performed in the context of the IST project IST-2004-511607 MobiLife, which is partly funded by the European Union. The authors would like to acknowledge the contributions of their colleagues, although the views expressed are those of the authors and do not necessarily represent the project.

References

[1] Dey, A.K., Abowd, G.D.: Towards a better understanding of context and context-awareness. Technical Report GIT-GVU-99-22, College of Computing, Georgia Institute of Technology (1999)
[2] Weiser, M.: Computer of the 21st century. Scientific American **265** (1991) 66 – 75
[3] Tennenhouse, D.: Proactive computing. Communications of the ACM **43** (2000) 43 – 50
[4] Hamilton, J.D.: Time Series Analysis. Princeton University Press (1994)
[5] Chatfield, C.: The Analysis of Time Series: An Introduction. 6th edn. Chapman & Hall (2003)
[6] Mayrhofer, R., Radi, H., Ferscha, A.: Recognizing and predicting context by learning from user behaviour. In: Proceedings of the International Conference on Advances in Mobile Multimedia (MoMM), Austrian Computer Society (OCG) (2003) 22 – 35
[7] Mayrhofer, R.: An architecture for context prediction. In: Advances in Pervasive Computing. Volume 176. Austrian Computer Society (OCG) (2004) 65–72

[8] Mayrhofer, R.: An Architecture for Context Prediction. PhD thesis, University of Linz (2004)
[9] Laasonen, K., Raento, M., Toivonen, H.: Adaptive on-device location recognition. In Ferscha, A., Mattern, F., eds.: Pervasive Computing, Proc. of. LNCS 3001, Springer (2004) 287–304
[10] Patterson, D.J., Liao, L., Fox, D., Kautz, H.: Inferring high-level behaviour from low-level sensors. In: Proceedings of the 5th International Conference on Ubiquitous Computing (UbiComp). (2003) 73 – 89
[11] Mozer, M.C.: The neural network house: An environment that adapts to its inhabitants. In: AAAI Spring Symposium on Intelligent Environments, AAAI (1998) 110 – 114
[12] Kaowthumrong, K., Lebsack, J., Han, R.: Automated selection of the active device in interactive multi-device smart spaces. In: Proceedings of the Ubicomp'02 Workshop on Supporting Spontaneous Interaction in Ubiquitous Computing Settings. (2002)
[13] Petzold, J., Bagci, F., Trumler, W., Ungerer, T., Vintan, L.: Global state context prediction techniques applied to a smart office building. In: Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS). (2004)
[14] P. Nurmi, P. Floréen, M.P., Linden, G.: A framework for distributed activity recognition in ubiquitous systems. In: in Proceedings of the International Conference on Artificial Intelligence. (2005) To Appear.
[15] Dey, A.K., Abowd, G.D., Salber, D.: A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. Human-Computer Interaction (HCI) **16** (2001) 97 – 166
[16] Nurmi, P., Przybilski, M., Lindén, G., Floréen, P.: An architecture for distributed agent-based data preprocessing. In: Proceedings of the Autonomous Intelligent Systems: Agents and Data Mining (AIS-ADM). (2005) To Appear.
[17] Flanagan, J.A., Mäntyjärvi, J., Himberg, J.: Unsupervised clustering of symbol strings and context recognition. In: Proceedings of the IEEE International Conference on Data Mining, IEEE (2002) 171–178
[18] Welsch, G., Bishop, G.: An introduction to the kalman filter. Technical Report TR 95-041, University of North Carolina (2004)
[19] Arulampalam, M., Maskell, S., Gordon, N., Clapp, T.: A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. IEEE Transactions on Signal Processing **50** (2002) 174–188
[20] Rabiner, L., Juang, B.: An introduction to hidden Markov models. IEEE ASSP Magazine **3** (1986) 4–16
[21] Flanagan, J.A.: Unsupervised clustering of symbol strings. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN), IEEE (2003) 3250–3255
[22] Rhodes, B.: Just-In-Time Information Retrieval. PhD thesis, Massachusetts Institute of Technology (2000)
[23] Wang, Y., Vassileva, J.: Bayesian network trust model in peer-to-peer networks. In: IEEE International Conference on Web Intelligence (WI). (2003) 372–378
[24] Buchegger, S., Boudec, J.Y.L.: A robust reputation system for mobile ad-hoc networks. Technical report ic/2003/50 epfl-ic-lca, EPFL, Lausanne, Switzerland (2004)
[25] Sprott, J.C.: Chaos and time-series analysis. Oxford University Press (2003)
[26] Russel, S., Norvig, P.: Artificial Intelligence: A Modern Approach. First edn. Prentice Hall (1995)