Intelligent Context-Awareness System using Improved Self-Adaptive Back Propagation Algorithm*

Sang-Hun Eo[1], Wei Zha[2], Byeong-Seob You[1], Dong-Wook Lee[1], and Hae-Young Bae[3]

Dept. of Computer Science & Information Engineering, Inha University
[1]{eosanghun, subi, dwlee}@dblab.inha.ac.kr, [2]zhazhago@hotmail.com and [3]hybae@inha.ac.kr

Abstract. Since the context plays a significant role in ubiquitous computing environment, many researches have studied about context-awareness system to improve the performance. An efficient learning mechanism is in importance of context-aware system, but there are seldom algorithms focused on convenience of systems by elaborating the learning mechanism with user's context information. As one of the most adaptable algorithm, Back Propagation provides us favorable inference capability. In this paper, we concentrate on improving the predict ability and reducing the system workload by proposing improved self-adaptive back propagation algorithm. The middleware we proposed improves the predicate capability. Thus, the overall performance becomes better than other systems. By adding system cache to middleware, it is possible for the context-aware system to act faster and improve the workload efficiency. Experiments show that there is an obvious improvement in overall performance of the context-awareness systems.

1 Introduction

According to the definition in [1], "Context is any information that can be used to characterize the situation of an entity". If a system uses context to provide the relevant information and/or services to its clients, it is context-aware. However, the relevancy depends on the user's task[1]. Context-awareness system collects, analyses and utilizes context. It is necessary for system to be context-awareness in ubiquitous environment[2].

As the popularity of sensor network and mobile terminal, mobility has become the main data source of context-aware environment. To provide better service to mobile users and avoid information flood, the context-awareness systems should be proactive according to the changing environment[3]. This requires context-aware systems to have the ability to learn users' environment and to predict users' behavior pattern. Most of the current computing systems are capable to process the users' explicit input and output the result. But they are not aware of users' status[4], for example, where is the user, what the user is doing, when the user does it, who is beside the user, and etc. Some of them focus on providing a uniform model to context, and some of them focus on providing an interface for different applications and distribute context, and some of them focus on how to manage context. But none of these systems can satisfy the requirement of ubiquitous computing environment completely, especially for the mobile user.

* This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment).

In this paper, we propose a new middleware for by using an advanced learning mechanism of context-aware systems. Back Propagation(BP) algorithm[10] is chosen as the basic learning mechanism due to its ability of learning relationships in complex data sets. The standard BP algorithm has limitations that it is too slow to provide the output in a short time. We introduced a dynamic momentum parameter added on the original BP algorithm with the view to make the BP algorithm avoid oscillations and respond instantly. The parameter value is calculated with reference to the LMS(Least Mean Square) between the obtained output and the desired output. By experiment, we prove that by using the improved BP algorithm in the proposed middleware, the system can precisely be proactive according to the users' changing environment.

This paper is organized as follows: in section 2, we briefly review the related work, previous context-awareness systems and the idea of primordial BP algorithm. In section 3, we propose an advanced learning mechanism SABPA (Self-Adaptive Back Propagation Algorithm) and an intelligent middleware for context-awareness system. In section 4, we make a comparison for the proposed middleware. The conclusion and future work are given in section 5.

2 Related Works

Most of the existing contxt-awareness systems have a common deficiency, that is, limitations in proactive and adaptation to the changing environment for the lack of learning mechanism. In [5], Dey et al. introduced their context toolkit system. The system aims to provide balanced touch to different kinds of applications through the distributed context and gives the idea how to achieve better performance, it is crucial for context-awareness systems to adopt proactive behaviors and the key point lies in finding user patterns. However, due to distinct characteristics of individual user behavior, user patterns are diversely different. Therefore, learning is considered as an efficient way to get the pattern.

In this chapter, our learning mechanism and improved BP algorithm for robust context-awareness systems are introduced. But it does not consider proactive and learning mechanisms. Roman et al. developed the Gaia context-awareness system to manage context[6]. This system throws light on assisting mobile applications to be developed and implemented in active spaces and supports mobile applications in a limited area. But it still lacks the ability of learning and proactive through the users' environment. A. shehzad et al. emphasized on interactivity among applications[7]. They proposed a context model in the purpose of having a common understanding of the contextual information but no proactive functions was mentioned. S.S.Yau et al. proposed the adaptive and object-based middleware RCSM in 2004[8]. This context-awareness system is designed for the mobile networks. So, it supports the mobile users and adapts to users' changing environment. In addition, this system supports using some specific contextual information to trigger corresponding actions. However, this system also lacks the ability of learning to proactive. H. Park et al. included context inference, learning, event triggering module of their middleware for the purpose of deal with various context[9]. In learning module, they listed some possible algorithms for learning but no material solutions.

For the robust context-awareness system, we chose BP algorithm as one of the most appropriate solution. From [10], a basic BP algorithm contains 3 layers: input layer, hidden layer and output layer. Input sample is processed step by step in terms of hidden

units through input layer. Then it is transmitted to output layer after passing through all the hidden units. During this process, the state of units of each layer influences the next layer. After comparison of present output to expected output, the results are transferred to BP module if they are not matched. During the BP process, error signals are transmitted through original forward path but in converse direction, and weight value is modified by each unit in each hidden layer on the expectation of the minimized error signals.

3 Context-awareness System using SABPA

To achieve better performance, it is crucial for context-awareness systems to adapt proactive behaviors and the key point lies in finding user patterns. However, due to distinct characteristics of individual user behavior, user patterns are diversely different. Therefore, learning is considered as an efficient way to get the pattern.

In this chapter, our intelligent context-awareness middleware and advanced learning mechanism SABPA for robust context-awareness systems are introduced.

3.1 Learning Middleware

We propose a new middleware that uses learning mechanism to provide useful context in context-awareness system. As shown in fig.1, we generally divide our middleware into three layers, namely, context source layer, context processing layer and context delivery layer. User and sensors can be the context sources (i.e. context can either come from user input or detected by the sensors).

Context Pre-process Module deals with the original context from users and sensors. Since the raw data are in variety of forms and expressions. It is necessary to convert the context into normalized forms which are easy for the system to process. Some context is aggregated to a higher level. We take GPS data for example; following is one data tuple of GPS:

*$GPGGA, 170647, 3726.4905, N, 12627.1471, E, 1, 03, 1.27, 2.7, M, -34.4, M, a*41*

We need to separate following information from this tuple:

Time = 17:06:47
Longitude- Latitude = (37° 26′ 49.05″, 126° 27′ 14.71″)
Altitude = 2.7m

And further, it (37° 26′ 49.05″, 126° 27′ 14.71″) is changed to specific location such as “Incheon International Airport” by Gauss projection. After that, useless data are abandoned here and essential context is sent to System Cache and stored in the context DB. The context DB stores the historical context including input context and the context predicted by the system.

Dynamic Adaptive Module between learning mechanism and context DB keeps the learning network up-to-date. The trigger event of update is controlled here, when new

arrival context accounts for certain percent of the total context, the old learning mechanism network can not predict accurately. We need to go on training our network to improve the accuracy. The Dynamic Adaptive module will continue training network with the new context in DB when the trigger event happened, until the error rate is within the default value.

Context DB inside the Dynamic Adaptive Module is the database of our middleware. It maintains and provides the data for training and update. Preprocessed data are sent here to store, when the trigger event happened, Dynamic Adaption Model will send message to ask Context DB provide the corresponding data to Learning Mechanism for training. After training, it is possible to categorize the new coming context to the already known classification and conclude the user pattern. By adjusting the default error rate, the accuracy can be controlled in an acceptable range. The structure of network such as number of hidden layers and nodes of each layer depends on applications specifically.

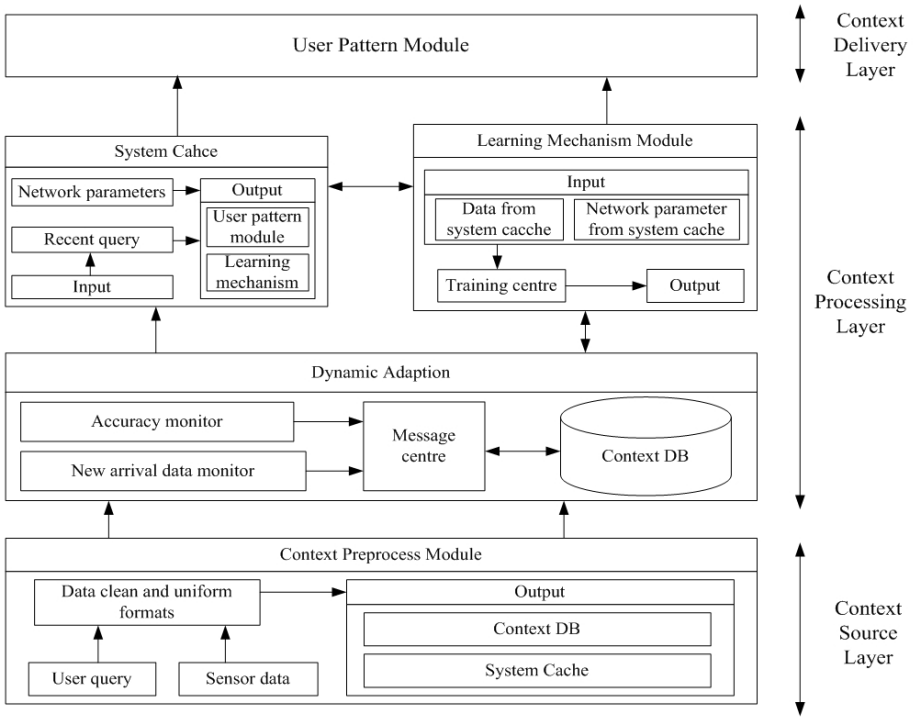

Fig. 1. Middleware for learning mechanism

System Cache makes context awareness system possible to response faster. Users would not change their habits suddenly; the maintaining of the latest input context and their options will dramatically improve the response time. These recorded options can be provided rapidly without computing when users repeat their queries. Also, the cache stores the parameters of learning network for different users (users use learning networks respectively, because each user is different individual), which makes it possible to support multiple users and response in a short time. Learning network, a background process that mines users' patterns, needs to be trained before using.

User Pattern Module stores the most recently query result. When the recently query is matched in System Cache, the corresponding answer can be provided directly from here. Learning Mechanism is the algorithm chosen for learning. In our research, an improved BP algorithm SABPA is introduced and will be discussed in next section

3.2 SABPA (Self-adaptive BP Algorithm)

The motivation of using BP algorithm is its ability of learning relationships in complex data sets which can not be easily perceived by humans. It has the ability to modify the output according to the changing of user context. It doesn't require the whole network rebuild when new context comes.

However, standard BP algorithm has two drawbacks. First, learning speed is slow. Traditional BP network can not provide fast response. Second, there are oscillations during learning. These problems cause its low efficiency. We have to improve the original BP algorithm so that it can be suitable for our proposing middleware. From [10], the formula for update weight in BP is

$$\Delta\omega_{ij}(t+1) = -\eta\partial E(t+1)/\partial\omega_{ij}(t) + \alpha\omega_{ij}(t)$$

Where η is the constant learning rate, which is usually chosen $0<\eta<1$ to guarantee that the weights space do not overshoot the minimum of the error space. α is momentum, which drastically affects the learning speed.

Table 1. SABPA algorithm

1. Initial η, $\omega(0)$, expect error ε and set k=1.
2. Set $\alpha(0) = \dfrac{\mathrm{E}(\omega(\mathrm{t}))}{\left\|\nabla \mathrm{E}(\omega(\mathrm{t}))\right\|^2}$, $\alpha = \alpha(0)$.
3. If $\mathrm{E}(-\eta g(\omega(t)) + \alpha\omega(t)) - \mathrm{E}(\omega(\mathrm{t})) \le -\dfrac{1}{2}\alpha\left\|\nabla \mathrm{E}(\omega(\mathrm{t})\right\|^2$ go to step 5; otherwise, set k=k+1 and go to step 4
4. Set $\alpha = \alpha(0)/m^{k-1}$ and go to step 3.
5. Set $\omega(t+1) = -\eta g(\omega(t)) + \alpha\omega(t)$ and $\alpha = \alpha(0)\dfrac{\mathrm{E}(t+1)}{\mathrm{E}(t)}$
6. If $\mathrm{E}(-\eta g(\omega(t)) + \alpha\omega(t)) \le \varepsilon$, then terminate; otherwise begin recursion.

m in step 4 is reduction factor. The choice of m value is not critical for successful learning, we choose m =0.5 in our algorithm.

E is the batch error measure defined as the sum-of-squared-differences error function over the entire training set

$$E = \sum_{k-1}^{m} E_k = \sum_{k}^{m} \sum_{t}^{q} (y_t^k - c_t^k)^2 / 2$$

By defining $g_i(\omega)$ as the gradient of $E(\omega)$ with respect to the *i*th variable ω_i, and $g(\omega)=(g_1(\omega),\ldots, g_n(\omega))$, the whole weights update can be written as

$$\omega(t+1) = -\eta g(\omega(t)) + \alpha\omega(t)$$

Also, $g(\omega)$ defines the gradient $\nabla E(\omega)$ of the sum-of-squared-differences error function E at ω.

Large values of momentum can accelerate the learning process but the drawback result is oscillations during learning. On reverse, smaller values of momentum decelerate the learning process. Normally, the momentum α is fixed before training. Here, we propose a strategy for dynamic adjusting momentum α ,called SABP (Self-adaptive BP Algorithm). This strategy makes the BP network faster and avoids oscillations. The basic idea is, check $E(t+1)$ every time. If $E(t+1) > E(t)$, which means learning is too slow, we should accelerate the learning process by increasing α. If $E(t+1) < E(t)$, which means learning is too fast, oscillations may occur, we should reduce the learning process by decreasing α. The proposed algorithm SABPA is illustrated in table 1.

When $E(-\eta g(\omega(t)) + \alpha\omega(t)) - E(\omega(t)) \le -\frac{1}{2}\alpha\left\|\nabla E(\omega(t)\right\|^2$, which means learning is a little fast, it is advantageous to slow down the learning process. Due to $\frac{E(t+1)}{E(t)} < 1$, then $\alpha(t+1) < \alpha(t)$. The learning process is decelerated.

When $E(t+1) > E(t)$, that means learning is too slow. It is necessary to increase the learning speed. Due to $\frac{1}{m^{k-1}} > 1$, then $\alpha(t+1) > \alpha(t)$ learning process is accelerated.

By dynamically adjusting momentum every time, SABPA avoids oscillations during learning and also accelerates the learning process.

3.3 Learning Steps

Because of the variety of raw data, some preparatory work needs to be done before learning, which are collection and cleaning of each user context, and storing them in context DB to train BP network. The raw data are first sent to Context Preprocess model which will clean the data and generate them to suitable format or higher level to make them available for proposed Learning Mechanism. Preprocessed data will be sent to Context DB which maintains all the necessary data for training and updating our network. According to different situation, diverse BP network (difference in number of hidden layers and nodes of each layer) will be constructed. The training of BP network

for each user should continue until the desired error rate is reached. Meanwhile the parameters of BP network for each user are stored in system cache respectively.

When user inputs query or sensors detect changing of environment, the context is sent to context pro-process module where contexts are converted into normalized forms and some context is aggregated to a higher level. The essential contexts are selected to be stored in the context DB and sent to the system cache. If the context information already exists in system cache, the output can be provided directly by system cache as it stores the recently queries and answers. Otherwise, the cache selects parameters of BP network according to the specific user and sends to BP network Learning Mechanism Model. In Learning Mechanism Model, BP network learns these contexts, categorize them to already known classification to fetch the user pattern and sends the results to both output and system cache for repeated queries. Fig 3 shows the learning steps.

The dynamic adaptation module is in charge of the update of SABPA network. When the accuracy of SABPA network decreases or new context takes up certain percent of the total context, this module sends messages to both Context DB and Learning Mechanism Model to manage them goes on training SABPA network until it matches the expected error rate. Thus, the accuracy is guaranteed.

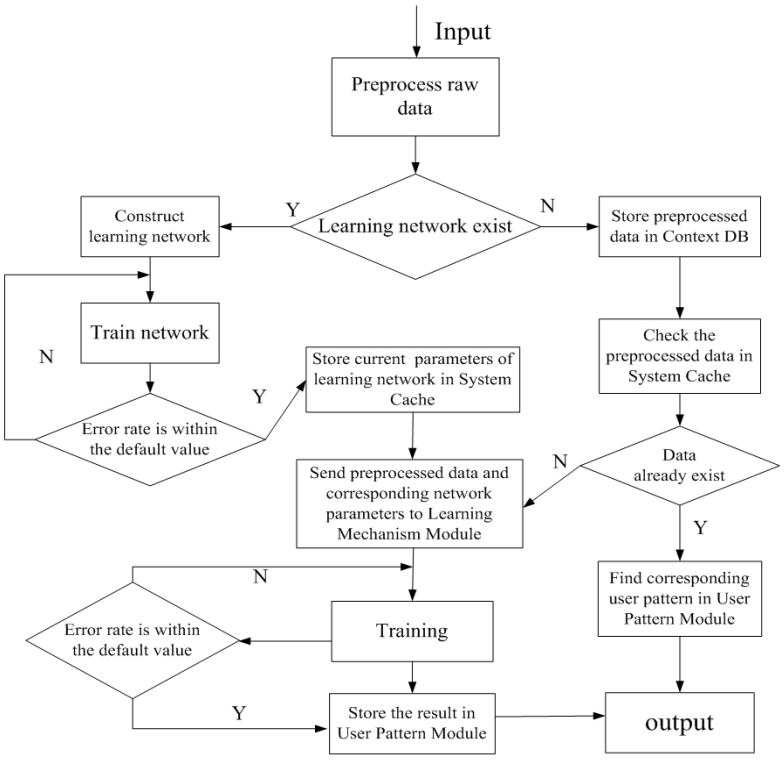

Fig. 3. Learning steps of proposed middleware

4 Implementation

We choose GPS system as the symbol of context-awareness system to implement our learning middleware. A basic BP network is generated by the Neural Network Toolbox for MATLAB for the learning process which contains three layers: input layer, hidden layer, and output layer). A user environment context DB which contains three months' context information is provided for training.

Table 3. Sample of training data set. Since the raw data is in variety of forms, we clean and normalize them to following forms

Input					Desired Output
User ID	Date	Time	Mode	From	To
User1	Monday	Morning	Bus	S1	C3,C4,C1
User2	Monday	Afternoon	Car	S1	C2,C1,C5
User1	Tuesday	Evening	Car	S3	C2,C3,C4
User3	Friday	Afternoon	Walk	S5	C6,C1,C2
User3	Wednesday	Morning	Car	S4	C5,C4,C5

We made a scenario using learning process to predict the usual destination of GPS user. Each GPS user has own pattern. We split the users into two halves depending on their behavior, one half of users with fixed routes, whose positions can be inferred accurately by their starting position and the other group of users with a high probability changing their routes frequently. For the first groups of users, we can make some explicit rule for inferring the destination. For the second group of users, it is difficult for human to explicit their rules. But if we can learn from these users' history DB, it is possible to get the user pattern to find the most likely destination in a high accuracy.

Table 4. Testing result. By using test data of table 3, our learning network output the most likely three destinations and their probability respectively.

Input					Output	
User ID	Date	Time	Mode	From	To	Probability
User 1	Monday	Morning	Car	S1	C1 C3 C4	60% 87% 74%
User 3	Friday	Afternoon	Walk	S1	C1 C2 C5	85% 64% 84%
User 1	Tuesday	Evening	Car	S3	C2 C3 C6	77% 80% 88%
User 4	Monday	Morning	Bus	S5	C2 C5 C6	91% 65% 46%
User 2	Thursday	Evening	Car	S4	C1 C3 C5	87% 69% 77%

In our scenario, the following context information is chosen: User ID, which distinguish different user; event date which illustrate the date when the event happens; event time, similar to the event date, illustrating the exact time when the event happens; transportation mode which gives the way how user behaves, say, where is the user from and where to go is illustrated by start position and destination.

We simply classify these contexts into input (User ID, event date, event time, mode, start position) and desired output(destination) for training the proposed BP network. Table 3 shows the training sample.

We train the SABPA network with context DB until we get the output within the desired error rate. Then, the network has the ability to give the most likely output.

In our case, according to the input context we create five nodes for input layer, six nodes for hidden layer, and three nodes in output layer for giving the most possible result. The testing result is shown in table 4.

5 Performance Evaluations

After building the learning network for context-awareness system, it is possible for the GPS system to predict the future location of a GPS user. Benefit from this ability, more information can be provided to the users without any query, such as weather, traffic jam, the shortest path. At the same time, workload is obviously reduced. Fig 4 shows the comparison of workload among systems.

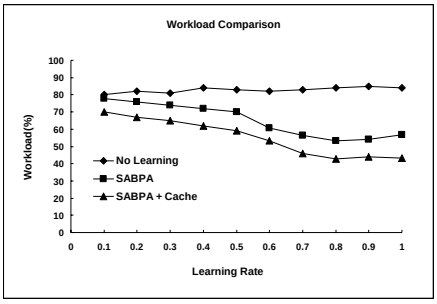

Fig. 4. The workload comparison between the system using learning and without learning

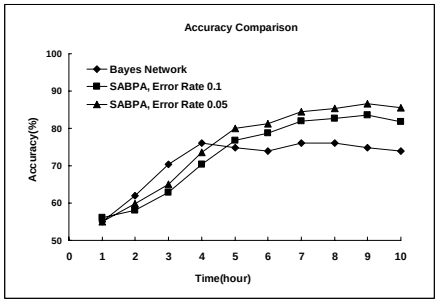

Fig. 5. The accuracy comparison between using Bayes and Backpropagation

Besides BP, other algorithms such as Baye's theorem are widely used. For GPS systems, Bayes could also be adopted as a satisfying learning mechanism. But BP is superior to Bayes on the control of prediction accuracy. Furthermore, when the new context accounts for some percent of the total context, Bayes network requires rebuilding rather than improvement on the current network. Fig 5 shows the accuracy comparison between Bayes and BP with different learning rates.

6 Conclusion and Future work

In this paper, we propose an intelligent learning middleware of context-awareness systems and using SABPA (Self-adaptive BP Algorithm) as learning mechanism for context recommend in ubiquitous computing environment. Different from standard BP, we improved it to become suitable for the proposed middleware regardless of its slowness. Using this middleware in our scenario, context-awareness systems evidently reduce the workload and provide better service. Through the tests, we proved that our middleware and SABPA performs far beyond just acceptable.

Future work in this are includes how to select the essential context for learning. Training a BP network is CPU-cost computing. Schedule management for training and user service is also important. In addition, the proposed SABPA is not yet completely optimized. More effective algorithm should be discovered for better performance.

References

1. A. K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. Technical Report GIT-GVU-99-22, Georgia Institute of Technology, College of Computing, June 1999.
2. A. Ranganathan and R. H. Campbell. A Middleware for Context-Aware Agents in Ubiquitous Computing Environments. In ACM/IFIP/USENIX International Middleware Conference 2003.
3. Byun, H.E., Cheverst, K.: Harnessing context to support proactive behaviours. In: ECAI2002 Workshop on AI in Mobile Systems, Lyon (2002)
4. Andreas Schmidt “Potentials and Challenges of Context-Awareness for Learning Solutions,” LWA 2005: 63-68.
5. A. K. Dey and G. D. Abowd. The context toolkit: Aiding the development of context-aware applications. In Workshop on Software Engineering for Wearable andPervasive Computing, Limerick, Ireland, June 2000.
6. Roman, Manuel, et al. Gaia: A Middleware Infrastructure to Enable Active Spaces, in IEEE Pervasive Computing, Oct-Dec 2002: 74-83.
7. A Shehzad, HQ Ngo, KA Pham, SY Lee.” Formal Modeling in Context Aware Systems,”International Workshop on Modeling and Retrieval of Context, 2004
8. S. S. Yau and F. Karim. An adaptive middleware for context-sensitive communications for real-time applications in ubiquitous computing environments. Journal of RealTime Systems, 26 (1): 29--61, 2004.
9. Hyunjung Park, “a middleware of context-awareness for ubiquitous computing middlewares” International Conference on Information Systems, Volume 00: Pages: 369 – 374,2005
10. Satish Kumar “neural networks” McGraw-Hill Education 2005