

## Context Selection and Reasoning in Ubiquitous Computing

Donghai Guan, Weiwei Yuan, Sungyoung Lee and Young-Koo Lee\*  
*Department of Computer Engineering, Kyung Hee University*  
{donghai, weiwei, sylee}@oslab.khu.ac.kr, yklee@khu.ac.kr

### Abstract

*Context reasoning refers to the process of giving high-level context deduction from a set of low-level contexts. It plays an indispensable role in ubiquitous computing. Most existing reasoning methods are proposed with the assumption that the knowledge of low-level context which is relevant to the given high-level context reasoning is available. When this information is lack, the existing methods blindly select some possible low-level contexts for reasoning, so that useless context might be included. These useless contexts have no or little favorable effect for reasoning and increase computation burden as well as repository burden. To deal with this problem, we use information gain-based method for context selection in our work. Only selected contexts are used for reasoning. Experimental results show that our proposed approach is promising.*

### 1. Introduction

One import factor that differentiates ubiquitous computing from traditional distributed computing is context. Context is any information that can be used to characterize the situation of an entity. An entity could be a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [1]. Context can be divided into low-level context and high-level context. In general, low-level context is simple and can be directly got from sensors or other sources. While, high-level context is abstract and need to be inferred from a piece of low-level context. High-level context can be as simple as taking in a name and returning the corresponding email address. It can also be more complex as taking in the number of people in a room, the relative gaze directions, the audio level, and the time of day, and returning whether or not a meeting was occurring. Most applications [2][3][4]

show more interest in high-level context. Hence, context reasoning plays an indispensable role for context-aware systems.

Many approaches have been proposed for context reasoning, including rule-based reasoning [5][6][7][8] and machine learning methods [9]. However, most context reasoning methods are proposed with the assumption that users have the knowledge about which kinds of low-level contexts are useful for a given high-level context reasoning. This knowledge is called “context relationship” in our work. In some cases, users do have this knowledge. However, context relationship is not always available, e.g. in mood reasoning, it is difficult for us to define which factors should be used for mood reasoning since it concerns many factors. When lack of information about context relationship, most reasoning methods blindly selects many possible low-level contexts for reasoning, so that useless context might be included. These useless contexts have no or little favorable effect on both rule-based reasoning and machine learning methods. Sometimes, they generate adverse effects. In addition to influence reasoning performance, using them also increase computation burden and repository burden.

In this paper, we use information gain-based method to solve the above problem. Experimental result shows that our approaching is promising.

This work is organized as follows. In section 2, we present the effects of context selection on reasoning. In section 3, the theory of information gain is presented. Section 4 shows the experiment result of our approach. At last, we present conclusions in section 5.

### 2. Context selection

Rule-based reasoning has proved itself an effective decision maker for many types of problems. However, the accuracy of such systems is highly dependent upon the accuracy of the user’s domain theory. When users learn or create a set of rules they are subject to a number of hindrances. As a result, the user-defined

---

\* Corresponding Author

rules are always incomplete or erroneous. In ubiquitous environment, the rules for high-level context reasoning might follow this format: If  $lc_1 = x_1$  and/or  $lc_2 = x_2$  ...and/or  $lc_n = x_n$ , then  $hc = M$  where  $lc_i$  ( $i=1,2,\dots,n$ ) represents low-level context and  $hc$  represents high-level context. It is easy to show that  $hc$  is determined by the value of  $lc_i$ . In ubiquitous environment, the number of different types of low-level context is huge. And it is hard for users to determine exactly which low-level context should be selected. As a result, many useless low-level contexts are mistakenly selected to deduce a given high-level context. To solve this problem, rules refinement is urgently needed.

Compared with user-defined rules, the accuracy of learning models utilizing machine learning methods is improved. However, one potential problem still exists. We take user mood as an example of high-level context. This high-level context deduction is treated as a classification problem since mood can be divided into several classes. Classification systems depend upon having the best set of input features from which a classification decision can be made. This is true both for the classifiers themselves and the learning models might be used to classify. This drives us to select the “relevant” low-level contexts for training, instead of all low-level contexts.

To achieve this function, firstly, we utilize information gain based method to find the most relevant low-level context for a given high-level context. Then, different inductive learning methods are used to deduce the learning models. Many work [10][11] showed that the learning models deduced from relevant features (low-level context) are much improved. In addition to refine rules and learning models, other advantages using information gain-based method include:

- Saving sensors. Only useful sensors need to be deployed since useless low-level contexts could be detected by using information gain-based method. As a result, the cost to build a smart environment is reduced.
- Saving context database size. Effectively mining useful low-level context can avoid data explosion.
- Decreasing computational burden. With the number of context decreases, the requirement of computational power is also decreased.
- Reducing reasoning uncertainty. Sensors have their inherent uncertainty. So the more sensors used for reasoning, the more uncertainty is arising.

The effects of context selection are shown in Fig. 1.

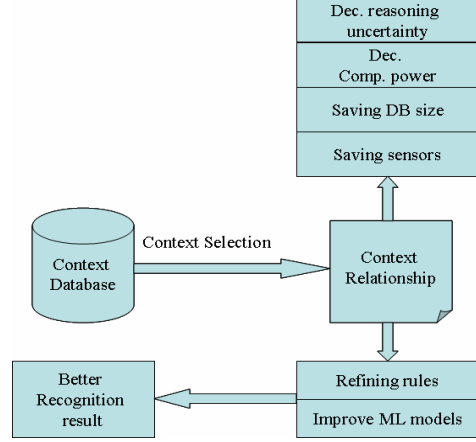


Figure 1. Effects of context selection

### 3. Information gain

In our work, context selection is determined by the weight of each low-level context. Instead of learning weights through a generic algorithm or other machine learning method, we use the information gain of each attribute, proposed by Qulilan [12], as its weight. The basic motivation for this study comes from the power of ID3 classification algorithm since ID3 is one of the best classifiers until now and it uses the concept of information gain as the criterion to select an attribute.

Let  $X$  be an attribute and  $C$  be the class variable. The following equations define the entropy of the class before and after observing the attribute respectively:

$$H(C) = - \sum_{c \in C} p(c) \log_2 p(c) \quad (1)$$

$$H(C|X) = - \sum_{x \in X} p(x) \sum_{c \in C} p(c|x) \log_2 p(c|x) \quad (2)$$

where  $x$  is a feature value and  $c$  is a class label. The amount by which the entropy of the class decreases after observing an attribute reflects the additional information about the class provided by that attribute and is called information gain:  $IG = H(C) - H(C|X)$ .

In other words, it measures how well a given feature separates the observed instances according to their given class categories. Without losing generality, suppose that there are  $N$  attributes (or features):  $x_1, x_2, \dots, x_N$ . Each attribute  $x_i$ ,  $i = 1, 2, \dots, N$ , is assigned a score based on the information gain over the class entropy due to observing itself:

$$IG_i = H(C) - H(C|x_i) \quad (3)$$

The ranking of the attributes is then done with respect to the values of  $IG_i$  in a descending order, reflecting the intuition that the higher an  $IG$  value, the

more information the corresponding attribute has to offer regarding the class.

Note that to compute the information gain, data sets with numeric features are required to be discretized. Many alternative methods can be applied for this. In the present work, we simply handle the discretization of continuous valued attributes by partitioning the range of values into a finite number of subsets, each of which has the same length.

#### 4. Experimental results

To test the feasibility of our reasoning engine, we make a set of experiments based on a publicly available context database (Provided by PERVASIVE 2004). This database includes information about human's activity and some low-level sensor data. These sensor data are used to infer human's activity through reasoning engine. In our experiments, 10 basic activities are considered. They are Lying, Kneeling, Sitting, Standing, Walking, Running, Climbing Stairs, Descending Stairs, Bicycling and Jumping. Totally 40 accelerometers are used in the experiments. They are strapped loosely to common trousers, 20 sensors per leg, starting from the ankle to the hip. Totally 2315 sample readings are used in our experiments. Two third of them are used as training set and the left as test set. Two machine learning methods are adopted in our experiment. They are back-propagation neural networks (BP) and k-nearest neighbors (KNN). In fact, many parameters in these two methods will influence the reasoning performance. For example, the node number of each hidden layer is an important parameter for BP network. Also, the number of hidden layers is important. As for KNN, K is the decisive parameter. How to set the parameters to achieve best classification result is beyond our discussion since our main purpose is comparing the reasoning result before information gain and after information gain. In our experiments, we construct a three layers BP neural networks for reasoning. The node number for the input layer is n, the low-level context number used for classification. The node number for the hidden layer is  $2*n+1$ . The output layer is 10, number of activities. As for KNN, we set  $K=5$ .

We first directly use BP networks and KNN to reason activities. Since there is no information gain-based feature selection function, all 40 sensors are used. The reasoning result is shown in table 1.

**Table 1.** Accuracy without information gain

	Feature Number	Accuracy (%)
BP	40	95.326

KNN	40	95.48
-----	----	-------

We then adopt information gain to sort the features according to their importance for classification in the following experiment.

To calculate information gain, the input must be discrete numbers. Since the inputs in our experiment are continuous real numbers, we handled the discretization of continuous valued attributes by partitioning the range of values into a finite number of subsets (subset number is 10), each of which has the same length. The information gains of these 40 sensors are shown in table 2.

**Table 2.** Information gains of 40 sensors

0.56	0.96	0.81	1.05	0.71
1.14	0.91	1.10	0.81	1.23
1.11	1.31	1.36	1.01	1.33
0.95	0.93	0.85	1.37	0.90
0.59	0.99	0.55	0.95	0.30
1.14	0.61	0.98	1.06	0.98
0.27	0.42	1.07	1.07	1.24
0.78	1.05	0.65	0.13	0.78

A feature's information gain is proportional with its weight to deduction. We can sort sensor's importance degree based on their information gains. Table 3 shows the sorted importance of each sensor.

**Table 3.** Sensors' weight ranking

19	13	15	12	35	10	26	6	11	8
33	34	29	4	37	14	22	28	30	2
24	16	17	7	20	18	3	9	36	40
5	38	27	21	1	23	32	25	31	39

We then perform a set of experiments using different sensor numbers. The result is shown in table 4. In this table, the left column is the feature number used in each experiment. For experiment when feature number is 1, it means only top 1 feature was used for reasoning, feature 19 in table 3. When feature number is 3, it means top 3 features (19,13,15) are used.

**Table 4.** Reasoning accuracies with different sensors

Features Number	BP Accuracy (%)	KNN Accuracy (%)
40	95.326	95.480
35	94.601	95.845
30	94.765	95.776
25	93.819	95.234
20	91.882	94.624
15	87.414	94.267
10	86.357	93.854
5	74.097	86.524
3	66.146	83.547
1	48.343	55.025

From the experiment result, we can see that when we use BP Neural networks to reason activity, the accuracy difference between 40 sensors and 20 sensors is 3.44%. It means the top 20 sensors can achieve 91.882% for reasoning, while, the other 20 sensors could contribute 3.44% based on it. If KNN is used for reasoning, the effect of information gain is clearer. When top ten sensors are used, the accuracy is 93.854, while, the accuracy is 95.48% for 40 sensors. The difference is only 1.63%.

## 5. Conclusions

In this paper, we utilize context selection mechanism before context reasoning. Information gain is used to filter out low-level contexts which do not contribute a lot for a given high-level context reasoning. We argue that our proposed context selection can refine human-defined rules and improve machine learning models. Simultaneously, it can decrease computational requirement and save repository size.

Our plan for future work is to assess our reasoning engine in our smart office. We are currently deploying different sensors in our office and will use them for collecting empirical data. Finally more high-level context will be inferred to further prove the feasibility of our approach.

## 6. Acknowledgment

This work is financially supported by the Ministry of Education and Human Resources Development (MOE), the Ministry of Commerce, Industry and Energy (Mocie) and the Ministry of Labor (Molab) through the fostering project of the lab of Excellency.

## 7. References

- [1] A.K. Dey, G.D. Abowd and D. Salber, "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications", *Journal of Human-Computer Interaction*, 2001, pp. 97-166.
- [2] A. Kulkarni, "Reactive Behavioral System for the Intelligent Room", Master Thesis, Computer Science and Engineering at the Massachusetts Institute of Technology, Cambridge, MA, 2002.
- [3] B.L. Brumitt, B. Meyers and J. Krumm, "EasyLiving: Technologies for Intelligent Environments", *Proceeding of Handheld and Ubiquitous Computing*, 2000, pp. 12-27.
- [4] S. Elrod, G. Hall and M. Dixon, "Responsive Office Environments", *Proceeding of ACM Communications*, 1993, pp. 84-85.
- [5] Gaia Project, <http://gaia.cu.uiuc.edu/>, 2005.
- [6] T. Gu, H.K. Pung and D.Q. Zhang, "A middleware for building context-aware mobile services", *Proceeding of IEEE Vehicular Technology Conference*, 2004, pp. 2656-2660
- [7] H. Chen, T. Finin and A. Joshi, "An ontology for context-aware pervasive computing environments", *Proceedings of the Workshop on Ontologies in Agent Systems*, 2003, pp. 197-207.
- [8] D. Ipina and E. Katsiri, "An ECA Rule-Matching Service for Simpler Development of Reactive Applications", *Proceedings of Middleware 2001 at IEEE Distributed Systems Online*, 2001.
- [9] N.Q. Hung and S.Y. Lee, "A Unified Middleware Framework for Context Aware Ubiquitous Computing", *Proceedings of the 2004 IFIP International Conference on Embedded And Ubiquitous Computing*, 2004, pp. 672-681.
- [10] H. Liu and Y. Lei, "Toward Integrating Feature Selection Algorithms for Classification and Clustering", *IEEE Transaction on Neural Networks*, 2002, pp. 143-159.
- [11] K. Nojun and C.H. Choi, "Input Feature Selection for Classification Problems", *IEEE Transaction on Neural Networks*, 2002, pp. 143-159.
- [12] J.R. Quinlan, "Induction of Decision Trees. *Machine Learning*", 1986, pp. 81-106.