J Electr Eng Technol Vol. 9, No. 3: 1051-1059, 2014
http://dx.doi.org/10.5370/JEET.2014.9.3.1051

ISSN(Print) 1975-0102
ISSN(Online) 2093-7423

Multi-Sensor Signal based Situation Recognition with Bayesian Networks

Jin-Pyung Kim*, Gyu-Jin Jang*, Jae-Young Jung and Moon-Hyun Kim†**

Abstract – In this paper, we propose an intelligent situation recognition model by collecting and analyzing multiple sensor signals. Multiple sensor signals are collected for fixed time window. A training set of collected sensor data for each situation is provided to K2-learning algorithm to generate Bayesian networks representing causal relationship between sensors for the situation. Statistical characteristics of sensor values and topological characteristics of generated graphs are learned for each situation. A neural network is designed to classify the current situation based on the extracted features from collected multiple sensor values. The proposed method is implemented and tested with UCI machine learning repository data.

Keywords: Structure learning, Bayesian networks, Multiple sensor signals, K2-learning algorithm, UCI machine learning repository

1. Introduction

In recent years, significant attention has focused on multiple sensor data fusion for context-aware and activity recognition applications. Data fusion techniques combine data from multiple sensors and related information to achieve more specific inferences than could be achieved by using a single, independent sensor [1]. As a result, in many applications, one can use more and more devices in the data fusion process. Furthermore, pushing the limit of hardware technologies is often hard and expensive for a given an application. Multi-sensor signal systems fuse the data measured and processed from many inexpensive devices [2].

Collecting data from different measurement devices has additional relationales. In many cases, systems built from a few but very high performance devices can be less robust than systems that use a large number of inexpensive devices and appropriate algorithms. Moreover, in some applications, such as sensor networks using multiple sensors can also provide users with crucial spatiotemporal information to exploit that one high performance measurement device alone cannot produce. In this context, multiple sensor signal algorithms and systems need to be developed to efficiently exploit a large amount of data collected using multiple sensors [5, 6]. In addition, faithful analysis of these multiple sensor signal algorithms and systems is also necessary to control the quality and cost of multiple sensor system.

† Corresponding Author: College of Information and Communication Engineering, Sungkyunkwan University, Korea. (mhkim@skku.edu)
* College of Information and Communication Engineering, Sungkyunkwan University, Korea. ({payon, gjjang}@skku.edu)
** Dept. of Computer and Information Warfare, Dongyang University, Korea. (jyjung@dyu.ac.kr)
Received: November 7, 2013; Accepted: December 30, 2013

There are approaches for situation recognition that equip the objects with Radio Frequency Identification (RFID) sensors. Buettner [7] evaluate RFID sensor networks for activity recognition, they prototyped a system that gathers object-use data in an apartment from WISPs (Wireless Identification and Sensing Platforms) and then infers daily activities with a simple Hidden Markov Model (HMM). Fused data from multiple sensors provides several advantages over data from a single sensor. First, if several identical sensors are used (e.g., identical radars tracking a moving object), combining the observations will result in an improved estimate of the target situation (position and velocity). A statistical advantage is gained by adding the N independent observations assuming the data are combined in an optimal manner. This same result could also be obtained by combining N observations from an individual sensor. A second advantage involves using the relative placement or motion of multiple sensors to improve the observation process. A third advantage gained by using multiple sensors is improved observability. Broadening the baseline of physical observables can result in significant improvements. A final advantage is accuracy improvement of recognizing situation by using a pattern extracted from multiple sensor signals [1].

In this paper, we propose intelligent situation recognition model from multiple sensor values. This paper has following contributions in multiple sensors based situation recognition studies. First, it shows new machine learning method for multiple sensor based recognition. The proposed learning method transforms collected multiple data to a graph, specifically a Bayesian network, and extracts common structural features from generated graphs for each situation. Second, it presents a new recognition method that (I) describes current situation as Bayesian network, (II) extracts structural features of generated Bayesian network and numerical features of nodes in Bayesian network to use them as situation description features and (III) propose a neural network consisting of input nodes which are grouped according to structural features of context networks. Each group of nodes performs merging of multiple sensor signals, since each input node represents an extracted value from a sensor signal.

In chapter 2, we present the proposed system. In chapter 3, we show experimental result by using robot failure data in UCI repository. Conclusion follows in chapter 4.

2. Multi-Sensor based Situation Recognition

The proposed approach consists of structure learning and class classification stage. Fig. 1 illustrates our situation recognition system. The Structure-Learning Stage performs i)quantization of failure data set by preprocessing part and ii) representing the current situation as a graph by analysis of multiple sensor signals, iii) extracting structure features from graph. The graph is a Bayesian network which is generated by using K2-algorithm from sampled multiple sensor data during a time window. The Bayesian network generalizes causal relationships between multiple sensor signals collected during non-overlapped time window. The graph is called as context-network. The structure feature expresses topological characteristics and numerical properties of nodes of context-network. The Classification Stage classifies situations as one of 4 classes based on the extracted structure features using neural network.

The recognition system works in 2 phases. The first phase is training phase, which generates context-networks from training set of each class. For each class, structural features are defined from generated context networks of this class. The structural features are used to design input nodes of the neural network. Structure-Learning Stage generates Context-Network (for four classes) to generalize causal relationships between multiple sensor signals collected during non-overlapped time window. The generated context-networks are analyzed to extract features for recognition of current situations. The second phase is classification phase, which learn and classify 4 classes

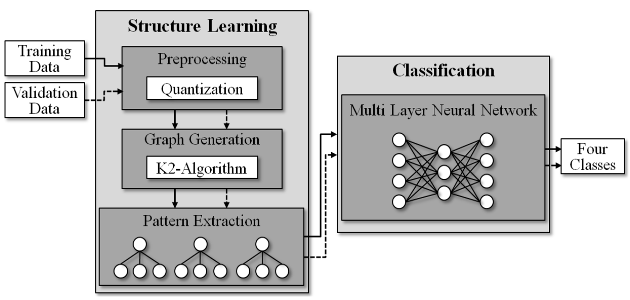


Fig. 1. Situation recognition system

(*Normal, collision, fr_collision, obstruction*) by extracted features. The extracted features are structural feature of context-network and numerical feature of context-network.

We define prototypical situations to build our model as 4 situations which frequently occur during the robot grasping activity. The number of situations can be easily extended without any theoretical modification of the proposed model.

2.1 Structure learning stage

Bayesian networks are probabilistic graphical models that provide interpretability of the explored domain by extracting and causal relationships between variables (sensors) representing the domain. The models readily combine patterns acquired from the data [3]. Generally, there are approaches of constraint-based and score-based, genetic methods for learning a Bayesian network structure [4]. We used K2-algorithm of the score-based structure learning algorithm to extract structural relationship between multiple sensor signals. The K2-algorithm proposed by Cooper and Herskovits [5, 8] is the most well-known Bayesian structure learning algorithms. The algorithm generates the Bayesian graph G with joint probability and Bayesian metric score. It is called K2-metric and is the most well-known Bayesian network evaluation function. K2-metric is expressed in the equation (1).

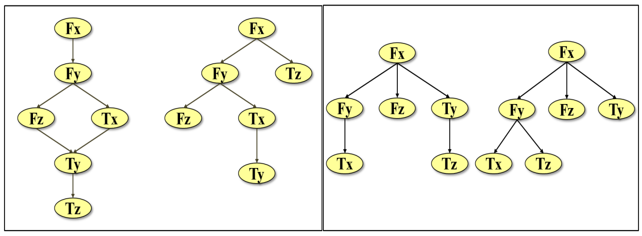
$$P(G,D) = P(G)\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}N_{ijk}\,! \qquad (1)$$

Maximizing $P(G,D)$ searches for the most probable Bayesian network structure G given a database D. $P(G)$ is the structure prior probability that is constant for each G. In equation (1), r_i represents the number of possible values of the node x_i. And q_i is the list of all possible instantiations of the combination. We let π_i as set of parents of node x_i.

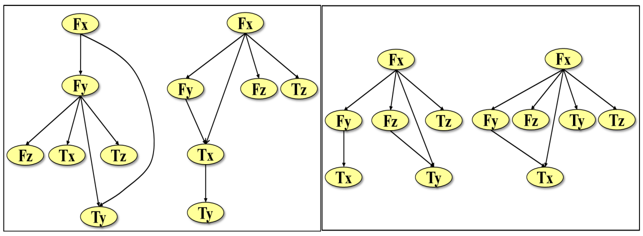
$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk}\,! \qquad (2)$$

N_{ijk} is the number of cases in D in which the attribute x_i is instantiated with its *k-th* value, and the parents of x_i in π_i are instantiated with the *j-th* instantiation in q_i. N_{ij} is the number of instances in the database in which the parents of x_i in π_i are instantiated with the *j-th* instantiation in q_i [5, 8].

The K2-algorithm starts by assuming that a node has no parents, after which, in every step it adds incrementally the parent whose addition mostly increases the probability of the resulting structure. K2-algorithm stops adding parents to the nodes when the addition of a single parent cannot increase the probability of the network given the data [8]. K2-algorithm statistically analyzes data, and data elements are representative of a Bayesian graph. Bayesian graph is a directed acyclic graph where directions of edges represent dependencies between nodes. Therefore, the relationship



(a) Normal networks (b) Collision networks



(c) Fr_Collision networks (d) Obstruction networks

Fig. 2. Generated context-networks of four classes

between the node and the node is represented by the dependence of the elements of the data expressed. Structure Learning Stage obtains the graphs for each of the four classes through the K2-algorithm from training data. These learned graphs are named as context-network G, and used to extract distinctive path patterns which are used as input features for each class recognition [11].

Typical context-networks for each class are shown in Fig. 2. Each context-network is directed acyclic graph, where each node represents a sensor signals (Fx, Fy, Fz, Tx, Ty, Tz). It is implemented as an adjacency matrix which represents a graph as a matrix of connections between nodes. The element of an adjacency matrix $A[i, j] = 1$ if there is an edge between *i-th* node and *j-th* node, or $A[i, j]=0$ otherwise. Context-networks for each class $c \in C$ are learned using its own training data. A set of context-networks for class $c \in C$ is named as G^C as equation (3), where $G_i^C \in G^C$ denotes *i-th* generated context-network of class C and α^c is the number of generated context-networks for class C.

$$\begin{aligned}
&\text{Context-Networks of Class } N \\
&\quad G^N = \{G_1^N, G_2^N, G_3^N \cdots G_{\alpha^N}^N\} \\
&\text{Context-Networks of Class } C \\
&\quad G^{Col} = \{G_1^{Col}, G_2^{Col}, G_3^{Col} \cdots G_{\alpha^{Col}}^{Col}\} \\
&\text{Context-Networks of Class } Fr \\
&\quad G^{Fr} = \{G_1^{Fr}, G_2^{Fr}, G_3^{Fr} \cdots G_{\alpha^{Fr}}^{Fr}\} \\
&\text{Context-Networks of Class } O \\
&\quad G^{O} = \{G_1^{O}, G_2^{O}, G_3^{O} \cdots G_{\alpha^{O}}^{O}\}
\end{aligned} \tag{3}$$

In the preprocessing stage, the collected data during non-overlapped time window through multiple sensors are uniformly quantized to get discrete values for each sensor. Quantization is needed to process a large number of finely changing data in the structure learning process. Without quantization various graphs which are not generalized form are generated depending on subtle variations of the data. For these graphs, K2-metric score is too low to be produced by the K2-algorithm as well as it is difficult to extract common patterns from them. So our approach eliminates a slight change of signals using uniform quantization for all sensor signals. The quantized *i-th* sensor signal sampled at $t=j$, $\hat{s}_j[j]$ is computed as

$$\hat{S}_j[j] = round\left[\frac{s_i[j]}{q_i}\right] \tag{4}$$

where $s_i[j]$ is *i-th* sensor value sampled at $t=j$, and q_i is quantization step size of *i-th* sensor signal. In proposed approach, we initialized quantization step size q_i to 10 for all sensors. In a context-network $G=(V,E)$ is a directed graph, where V is a set of nodes and E is a set of edges. An edge $e=\langle n_s,n_e\rangle \in E$, where n_s, n_e are tail and head of edge e, respectively, represents causal relationship, that is n_s affects occurrence of n_e. Thus structural features which are topological characteristics of the generated context-network reflect these causal relationships among nodes in current situation. The path of the generated graphs in structure learning stage indicates patterns, which describes specific relations between each sensor node characterizing each class. The path patterns from generated context-network are extracted, and these are used as structural features for situation recognition. Fig. 3 depicts the process of extracting patterns from the context-network. P_{ij}^N denotes *j-th* path of G_i^N, which is *i-th* context-network of the normal class. $K_{n,i}$ is the number of extracted path

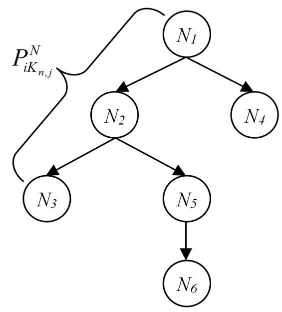


Fig. 3. Path pattern extraction from context-network

Table 1. Path pattern from bayesian graphs of normal class

G^N				
G_1^N	G_2^N	G_3^N	• • •	$G_{\alpha^N}^N$
P_{11}^N	P_{21}^N	P_{31}^N	• • •	$P_{\alpha^N 1}^N$
P_{12}^N	P_{22}^N	P_{32}^N	• • •	$P_{\alpha^N 2}^N$
⋮	⋮	⋮	⋮	⋮
$P_{1K_{n,1}}^N$	$P_{2K_{n,2}}^N$	$P_{3K_{n,3}}^N$	• • •	$P_{\alpha^N K_{n,j}}^N$

patterns from G_i^N. Each path pattern is a path from root to the leaf node of context-network [17]. Each path is represented as a sequence of nodes ordered from root node to the leaf node. In Fig. 3, 3 paths from root to the leaf nodes in the context-network are extracted, i.e. N_1-N_2-N_3 and N_1-N_2-N_5-N_6, N_1-N_4 where N_i is the *i-th* node.

Table 1 shows path patterns $P^N=\{P_{i1}^N, P_{i1}^N, \ldots P_{iK}^N\}$, $i=1,2,\ldots\alpha^N$ for the context-networks of normal class. For collision class, fr_collision class and obstruction class, we generate path patterns P^{Col}, P^{Fr} and P^O, respectively from the learned context-networks.

2.2 Classification stage

Classification stage classifies input pattern into four classes. We designed a 2-layer neural network for pattern classification using extracted path patterns. Neural networks have emerged as an important tool for classification. The recent vast research activities in neural classification have established that neural networks are a promising alternative to various conventional classification methods. The advantage of neural networks lies in the following theoretical aspects. First, neural networks are data driven self-adaptive methods in that they can adjust themselves to the data without any explicit specification of the functional or distributional form of the underlying model. Second, they are universal functional approximations in that neural networks can approximate any function with arbitrary accuracy [12, 13]. For each class $c \in C$, the occurrence probability of each path $p \in P^S$ during training, $P(p)$ is computed. For each class, the path patterns with highest probabilities are selected and used as input features of classification [17, 19]. These selected path patterns are defined as path features. For each distinctive selected path features $p_i=N_{i1}$-N_{i2}-$\ldots N_{im}$, a set of input nodes of neural network $IP_i=\{N_{i1},N_{i2},\ldots N_{im}\}$ is assigned. Every node in p_i, i.e. $N_{ij}, j=1,2,\ldots,m$ is assigned as an input node of IP_i. Table 2 shows path features which are selected path patterns for each class. The input nodes in each selected path feature are grouped separately as shown in Fig. 5. Note that a path feature can appear in different classes repeatedly, i.e. IP2 in collision class and obstruction class. The *null* denotes that there is no such path pattern in the context-network of that class. By using this organization of input layer, we can reflect not only topology of the generated context-network but also numerical properties of sensor signals. The output layer of neural network consists of 4 nodes, each of which corresponds to a class. The weights of connections are learned using back-propagation algorithm. During training phase, the input nodes belonging to the path features for the target class are given preprocessed current input values while other input nodes are given 0's. During classification phase after training, the input nodes of P_i is provided with preprocessed sensor signals if generated context-network has path P_i, otherwise they are given 0. The current class is classified as

$$c_c = \arg\max_{C} v(o_c) \tag{5}$$

where o_c is output node for a class c, and v(.) denotes the value of an output node. Fig. 4 shows the whole process of the second phase for test and validation. In Fig. 4(a) shows quantization of a test dataset, Fig. 4(b) shows generation of a context network, Fig. 4(c) shows extraction of 3 path patterns from the generated context network, and Fig. 4(e) shows providing numerical values of sensors included in each path as input values of neural network.

In our proposed method, accordingly; First) classification stage must have to use a neural network, and Second) structure learning stage is necessary to provide a neural network with the input values from the extracted path patterns. Third) we perform classification for the newly entered situation based on the sensor data collected during time window.

Table 2. Define path feature of classes for input of neural network

Path Feature \ Classes		Normal	Collision	Fr_Collision	Obstruction
IP_1	1st Node	IP_1^N	*null*	IP_1^F	*null*
	2nd Node				
IP_2	1st Node	*null*	IP_2^C	*null*	IP_2^O
	2nd Node				
	3rd Node				
⋮	1st Node	⋮	⋮	⋮	⋮
	2nd Node				
IP_n	1st Node	IP_n^N	*null*	IP_n^F	IP_n^O
	2nd Node				

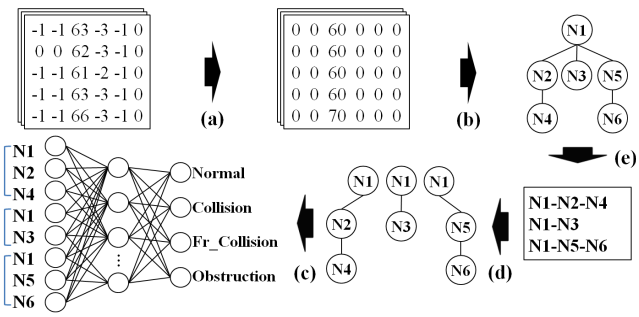


Fig. 4. Process of validation phase

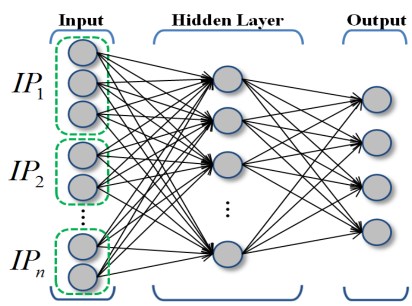


Fig. 5. Neural network architecture

3. Experiments and Evaluation

By using UCI Machine Learning Repository dataset, we have fully performed experiments to extract patterns from signals that are collected from multiple sensors in our proposed structure, and have utilized the input of the neural network of both extracted patterns of structural property and representing value of patterns in the classification stage. Through such experiments, we could analyze data collected from multiple sensors, and had performance verification how to recognize the situation. Moreover, we have carried out a comparison with objective performance of MLP and Bayesian Network Classifier (BNC), our proposed method.

3.1 Multi-sensor signals for context-awareness

We used robot execution failure data from the UCI Machine Learning Repository for conducting our experiments. Dataset is a collection of failure data occurring in approach process to grasp position. And dataset contains force and torque measurements of robot sensors during operation [10, 11]. Four situations of system behavior, that will be learned, are considered: i.e. *normal, collision, front collision* and *obstruction*. A *normal* situation represents a situation that a robot moves to a grasp position without any problems. In a *front collision* situation, there exists a collision between an obstacle and a front part of a robot. In a collision situation, there is a collision between an obstacle and the other part of the robot. The *Obstruction* represents a situation that robot is clogged with an obstacle. Each data set in a window D_t is constructed from sampled values of multiple sensors during fixed time interval of 315ms. A data set $D_t=(T[1], T[2],\dots,T[n])^T$ is a sequence of sampled data sets, where $T[i]$ is a set of sampled data set at $t=i$, and $n=15$ is window size.

$$T[i] = \{F_x[i], F_y[i], F_z[i], T_x[i], T_y[i], T_z[i]\} \tag{6}$$

Consists of three sampled force data for each axis, *Fx*, *Fy* and *Fz*, and three sampled torque data for each axis, *Tx*, *Ty* and *Tz*. We use 88 window datasets in this experiment. Table 3 shows the configuration of a window dataset. In this dataset, there are 15 sampled datasets [10, 11]. Table 3 shows an example of the distribution of sensor values of D_t

Table 3. Data in a window of normal class

Sensors / Time	*Fx*	*Fy*	*Fz*	*Tx*	*Ty*	*Tz*
1	-1	-1	63	-3	-1	0
2	0	0	62	-3	-1	0
3	-1	-1	61	-3	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	-1	0	64	-2	-1	0

for normal class. Note the different ranges depending on sensors and changes of sampled values with respect to the time.

3.2 Structure learning from multi-sensor signals using K2-algorithm

Robot execution failure Dataset is collected during regular time by observation window. Each failure is characterized by 15force/torque samples collected at regular time intervals starting immediately after failure detection [9, 10]. Fig. 6 shows change of 6 sensor data in a data window after failure detection. It is observed that not only the distribution of each sensor values depends on class, but also the joint distribution of pair of sensor values differs according to the class. Because of such properties, it is necessary to extract structural properties of the data by learning algorithms [16, 20].

Since using one data window is not sufficient to describe current class, we constructed a 3-Cut data as an input of the Structure Learning Stage by concatenating 3 Data Windows. Also, we have formulated a training set that consists of 42 3-Cut data out of 88 data windows. Therefore the Structure Learning module learned 42 context-networks of the normal class, one for each 3-Cut data. Each of context-networks has multiple paths from root node to terminal nodes as shown in Table 4. Patterns occurring frequently in context-network represent particular correlations between sensor signals that we have to treat significantly [21]. The most frequently appeared patterns are shown in the Table 4(b). We can see that a *1-2-4-5-6 pattern* appeared most frequently in the normal class. Also in Table 4(c), the values of each nodes included in the same path pattern has similar values for the generated context-networks for normal class.

Table 5 shows extracted path patterns for 4 classes. Proposed method extracts the most frequently occurring 4 path patterns (which are shown in shaded boxes) for each class. The number of patterns in Table 5, such as 6 in *1-2-4-5-6 path* pattern in the normal class, denotes the number of context networks of normal class which include *1-2-4-5-6 path* during training phase. It is observed that less frequently appearing path patterns degrade accuracy of recognition [22].

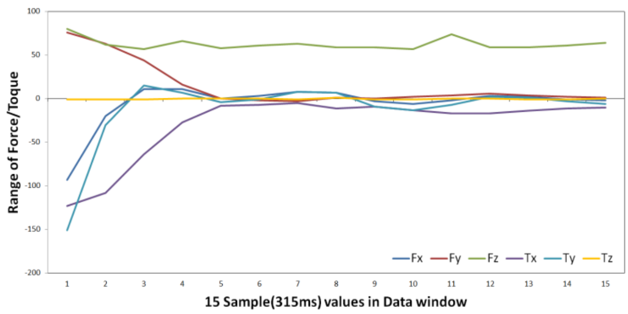


Fig. 6. Change of sensor values in a data window

Table 4. Extracted path patterns from normal context-network

(a) Path pattern from normal class

Normal Context-Network G^N						
G_1^N	G_2^N	G_3^N	G_4^N	G_5^N	G_6^N	G_7^N
1-2-3-4-5-6	1-2-4	1-2-4-5-6	1-2-3	1-2-4	1-2-3-5-6	1-2-3
	1-3	1-3	1-2-4	1-3-4	1-2-4-5-6	1-2-4-5
	1-5-6		1-5-6	1-5-6		1-6
G_8^N	G_9^N	G_{10}^N	G_{11}^N	G_{12}^N	G_{13}^N	G_{14}^N
1-2-4-5-6	1-2-3	1-2-4-5-6	1-2-4	1-2-3	1-2-3	1-2-3-4-5
1-3	1-2-4-5-6	1-3	1-3	1-2-4	1-2-4-5-6	1-2-3-5
			1-5-6	1-2-5-6		1-6

(b) Most frequently appeared path patterns

1-2-4-5-6	1-3	1-2-4	1-2-3	1-5-6	1-6
6 patterns	5 patterns	5 patterns	5 patterns	4 patterns	2 patterns

(c)Value of each node in path patterns

1-2-4-5-6 Pattern						
Graph / Node	G_3^N	G_6^N	G_8^N	G_9^N	G_{10}^N	G_{13}^N
1	0	0	0	0	0	0
2	0.22	0	0.22	0	0.22	0
4	-10	-10.2	-5.11	-8.89	-7.11	-10.9
5	-2.89	-3.56	-0.89	-2.89	-1.78	-3.78
6	0	0	0	0	0	0

Table 5. Path patterns of each class

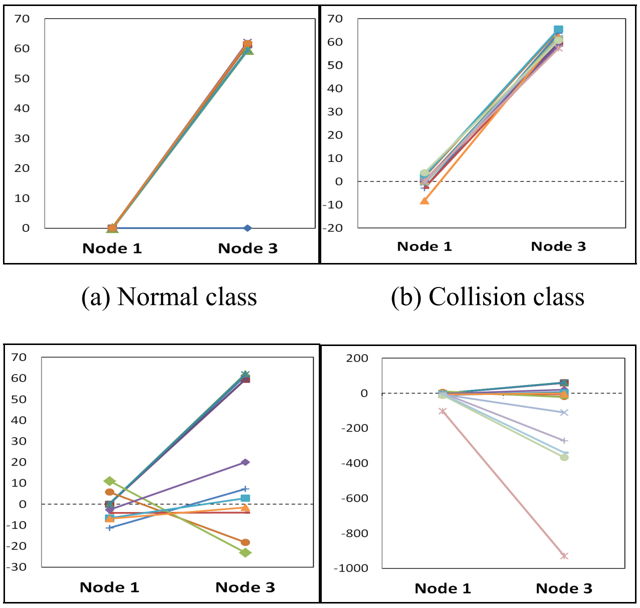
Normal	1-2-4-5-6	1-3	1-2-4	1-2-3	1-5-6	1-6
	6	5	5	5	4	2
Collision	1-2-4	1-5	1-3	1-2-6	1-2-3	1-5-6
	15	12	10	8	3	3
Fr Collision	1-2-3	1-3	1-5	1-6	1-4	1-2-4-5-6
	11	11	9	8	6	4
Obstruction	1-6	1-5	1-2-3	1-3	1-2-5	1-4
	23	20	16	15	13	12

3.3 Pattern classification

The input data to be classified are applied to structure learning stage to construct a context-network C_{in}. From C_{in}, all paths from root node to leaf nodes l_i, $i=1,2,\ldots,n$ are extracted. For each path l_i, it is checked to see it this path exists in the input layer of neural network. It this path exists as IP_i, the sensor values of the nodes in l_i are applied to the corresponding nodes in IP_i. As shown in Table 4, some path patterns, for example *1-3 pattern*, exist in context-networks of different classes. However, their sensor values of nodes in the path are different depending on the class. Fig. 7 shows conditional joint probability distribution of *node1* and *node3*,

$$P(node1, node3 \mid C), \quad C \in \{N, Col, Fr, O\} \tag{6}$$

Fig. 7(a) shows *P(node1, node3|N)*, in which *node1* mostly has value 0, and *node3* mostly has value 60. Although *node1* has mostly value 0 in *P(node1,node3|O)*



(a) Normal class (b) Collision class

(c) Fr_Collision class (d) Obstruction class

Fig. 7. Sensor data distribution of each class

as shown in Fig. 7(d), *node3* has quite different distribution compared with *P(node1, node3|N)*. While the values of *node3* have similar distributions between normal class and collision class as shown in Figs. 7(a) and Fig. 7(b), *node1* has different distributions of values in these two classes. In fr_collision class and obstruction class, *node3* has wider distribution of values than normal class and collision class. We can notice the value of *node3* is generally less than or equal to the value of *node1* in obstruction class. On the contrary, *node1* has smaller value than *node3* both in normal class and collision class. We reflect these characteristics of sensor value distributions with respect to classes into structural features. The strong correlation between nodes causes existence of an arc between these nodes in context-network.

Table 6 shows 14 data used as training set of normal class. The green shaded boxes represent average values of each sensor node for four path patterns defined for the normal class, i.e. *1-3 path*, *1-2-3 path*, *1-2-4 path*, and *1-2-4-5-6 path*. Table 7 shows performance comparison between the proposed methods and 2-layer perceptron. Input values for two layer perceptron are provided in two ways. The first method uses average value of sensor signals during concatenated time window.

$$sensor_i^{avg} = \frac{1}{45}\sum_{k=1}^{3*15} sensor_i[k], \quad i = 1,2,\ldots,6 \tag{7}$$

The second method computes sensor value as average of maximum as shown in equation (7).

$$sensor_i^{rep} = \frac{1}{3}\sum_{k=1}^{3} \underset{j}{Max}\{sensor_i^k[j]\} \tag{8}$$

Table 6. Input value of neural network for normal state

Path Features \ Training Data		Normal													
		T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9	T_{10}	T_{11}	T_{12}	T_{13}	T_{14}
1-3 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 3	0	0	0	0	0	0	0.22	0	0	0	0	0	0	0.22
1-5 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1-6 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1-2-3 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 2	0	0.22	0.22	-0.22	-0.44	0	0.22	0.22	0	0.22	-0.67	0	0	0.22
	Node 3	61.56	61.33	59.56	61.11	61.56	59.33	59.33	62.22	60.67	59.56	61.78	60.89	59.56	59.11
1-2-4 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 2	0	0.22	0.22	-0.22	-0.44	0	0.22	0.22	0	0.22	-0.67	0	0	0.22
	Node 4	-2.67	-8.44	-10	-3.11	-7.33	-10.22	-12	-5.11	-8.89	-7.11	-5.33	-8.44	-10.89	-8
1-2-5 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1-2-6 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 6	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1-2-4-5-6 Path	Node 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	Node 2	0	0.22	0.22	-0.22	-0.44	0	0.22	0.22	0	0.22	-0.67	0	0	0.22
	Node 4	-2.67	-8.44	-10	-3.11	-7.33	-10.22	-12	-5.11	-8.89	-7.11	-5.33	-8.44	-10.89	-8
	Node 5	-0.44	-2.22	-2.89	-0.66	-2.22	-3.56	-5.56	-0.89	-2.89	-1.78	-1.33	-2.89	-3.78	-4
	Node 6	0	0	0	0	0	0	0.22	0	0	0	0	0	0	0.22

Table 7. Performance evaluation of situation recognition

	Multi-Layer Perceptron				Bayesian Classifier		Proposed Method	
	Average		Representative		Average	Representative	K2	
Training Rate (Hidden Node)	70% (2)	70% (10)	70% (23)	90% (12)	70%	70%	80% (40)	80% (50)
Precision Average	96%	96%	98%	74%	80%	89%	98%	100%

In equation (8) $sensor_i^k[j]$ stands for *j-th* value of $sensor_i$ in *k-th* data window. The performance is evaluated using average of recognition precision as following.

$$\frac{1}{4}\sum_{Class_i} precision_i = \frac{1}{4}\sum_{Class_i} \frac{\textit{number of True Positives}}{\textit{total number of classes classified as class}_i} \tag{9}$$

We experiment with changing the portion of the training data and the number of nodes in the hidden layer. In Table 7, the third row shows the portion of training data out of whole data, and the number of nodes in the hidden layer as a number in parentheses used in each experiment. The fourth row denotes average of recognition precision for each experiment. It shows average precision of the proposed method becomes large as the number of hidden nodes grows. Generally, conventional MLP shows better result when average input ($sensor^{avg}$) is used than average of maximum input ($sensor^{rep}$) is applied. However, if we provide large number of hidden nodes in case of using $sensor^{rep}$, it shows good performance (98%). When we use $sensor^{avg}$ in conventional MLP, the number of hidden nodes does not affect precision. Also, conventional Bayesian classifier shows better result when average input ($sensor^{avg}$) is used than average of maximum input ($sensor^{rep}$) is applied. Generally, the proposed method shows better performance compared with conventional MLP and Bayesian classifier. The best performance is acquired when we use 50 hidden nodes and 80% of whole data as training data.

Fig. 8 shows comparison of the confusion matrices for the proposed method and multilayer neural networks, Bayesian classifier. Each column denotes ground truth, while each row represents system truth. The confusion matrices for the proposed method with different number of hidden nodes, i.e. 50 and 40, are shown in Figs. 8(a) and (b), respectively. In both matrices in (a) and (b), only Fr_collision class results in one incorrect classification as collision class. And Fig. 8(c) (d) shows the confusion matrices for result of the compared method, based on MLP with average sensor value and representative value as input. The Fr_collision class is misclassified to collision class in 2 test cases in Fig. 8(c). Two test cases of collision class are misclassified to Fr_collision class in Fig. 8(d). And, Figs. 8 (e) (f) shows the confusion matrices for Bayesian classifier using representative value and average value of each sensor in dataset as an input of MLP. Fig. 8(e) shows poor performance especially for collision and Fr_collision

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	14	0	0	0
Collision	0	10	0	0
Fr_Collision	0	0	10	0
Obstruction	0	0	0	11

(a) K2 80%(50)

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	14	0	0	0
Collision	0	10	1	0
Fr_Collision	0	0	9	0
Obstruction	0	0	0	11

(b) K2 80%(40)

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	14	0	0	0
Collision	0	10	2	0
Fr_Collision	0	0	8	0
Obstruction	0	0	0	11

(c) MLP(average) 70%(2)

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	21	0	0	0
Collision	0	17	2	0
Fr_Collision	0	0	14	0
Obstruction	0	0	0	34

(d)MLP(representative) 70%(23)

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	20	1	0	0
Collision	2	13	1	1
Fr_Collision	1	1	10	4
Obstruction	0	1	5	28

(e) BC(average) 70%

System Truth \ Ground Truth	Normal	Collision	Fr_Collision	Obstruction
Normal	20	0	1	0
Collision	0	15	0	0
Fr_Collision	0	0	13	3
Obstruction	0	1	2	21

(f) BC(representative) 70%

Fig. 8. Confusion matrix for proposed method and MLP, bayesian classifier (BC)

classes. While Fig. 8 (e) shows relatively better result than Fig 8. (f). It can be noticed that overgeneralization degrades performance as shown in Fig. 8 (e).

4. Conclusion

In this paper, we focus on studying the recognition of multiple sensor signals. We propose a systematic structure learning approach to automatically learn the Context-Network. Neural networks using structural features (pattern) is designed and show this approach can achieve improved classification performance. Specifically, our structure learning process consists of three stages: sensor data quantization stage, the subsequent context-network generation stages with K2-algorithm and the path pattern extraction stage from context-network. Our automatically learned situation recognition model outperformed the Multi-Layer Perceptron on robot execution failure datasets. These results demonstrate the feasibility and recognition accuracy of the proposed approach for multiple sensor signals. In the future, the study to expand path features for improving class recognition performance is required. Also, automatic generation of path features from context-networks should be investigated. The proposed method can be directly applied to multi-sensor based recognition system such as surveillance system, or context-aware system.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2012R1A1A2008998).

References

[1] David L. Hall, James Llinas, “Multisensor Data Fusion”, ISBN 0-8493-2379-7, 2001, CRC Press LLC

[2] R. C. Baker and B. Charlie, “Planning and acting in partially observable stochastic domains”, Artificial Intelligence, Volume 101, Issues 1-2, pp. 99-134, 1998.

[3] Lerner, Boaz, and Roy Malka*, “Investigation of the K2 algorithm in learning Bayesian network classifiers.”, Applied Artificial Intelligence 25.1 (2011): 74-96.

[4] Chen, Xue-Wen, Gopalakrishna Anantha, and Xiaotong Lin, “Improving Bayesian network structure learning with mutual information-based node ordering in the K2 algorithm.”, Knowledge and Data Engineering, IEEE Transactions on 20.5 (2008): 628-640.

[5] Gregory F. Cooper and Edward Herskovits, “A Bayesian method for the induction of probabilistic networks from data”, MACHINE LEARNING, Volume 9, Number 4, 309-347,1992.

[6] J. Ramon and L. De Raedt, “Multi instance neural networks”, Workshop on Attribute-Value and Re-lational Learning, In Proceedings of ICML-2000, 2000.

[7] Michael Buettner, Richa Prasad, Matthai Philipose, David Wetherall, “Recognizing Daily Activities with RFID-Based Sensors”, ACM International Joint Conference on Pervasive and Ubiquitous Computing, (2009), 51-60.

[8] Evelina Lamma, Fabrizio Riguzzi and Sergio Storari, “improving the K2 Algorithm Using Association Rule Parameters”, Modern Information Processing, 207-217, 2006A.

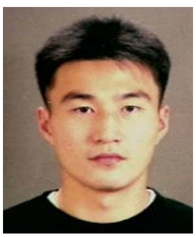
[9] B. Horling, R. Vincent, R. Mailler, J. Shen, R. Becker, K. Rawlins, V. Lesser, “Distributed Sensor Network for Real Time Tracking”, Proceedings of the 5th Inter-national Conference on Autonomous Agents (2001) 417-424.

[10] UCI repository of machine learning databases, Robot Execution Failures Data Set http://archive.ics.uci.edu/ml/datasets/Robot+Executio n+Failures

[11] Lopes, Luis Seabra, and Luis M. Camarinha-Matos. “Feature transformation strategies for a robot learning problem.” Feature Extraction, Construction and Selection, Springer US, 1998. 375-391.

[12] Guoqiang Peter Zhang, “Neural Networks for Classi-fication: A Survey”, IEEE transactions on systems,

man, and cybernetics, Vol. 30, No. 4, November 2000.
[13] Daniel Roggen, Martin Wirz and Gerhard Troster, "Recognition of crowd behavior from mobile sensors with pattern analysis and graph clustering methods", Networks and Heterogeneous Media, American Institute of Mathematical Sciences, Volume 6, Number 3, September 2011, pp. 521-544
[14] K. Bao and S. Intille, "Activity Recognition from User-Annotated Acceleration Data", Proc 2nd Int Conf Pervasive Computing, (2004), 1-17.
[15] J. A.Ward, P. Lukowicz and H. Gellersen, "Performance metrics for activity recognition", ACM Transactions on Information Systems and Technology, 2 (2011), 6:1-6:23.
[16] Christopher R. Wren, Emmanuel Munguia Tapia, "Toward scalable activity recognition for sensor networks", Lecture Note in Computer Science, Springer-Verlag, pp. 168-185, 2006.
[17] Tae-Ki An, Moon-Hyun Kim, "Context-Aware Video Surveillance System", Journal of Electrical Engineering & Technology, Volume 7, Number 1, 115-123, 2012.
[18] P. Langley, W. Iba, K. Thompson, "An analysis of bayesian classifiers", in: Proceedings of the Tenth National Conference on Artificial Intelligence, AAAI Press, San Jose, CA, 1992.
[19] K. Y. Eom, J. Y. Jung, and M. H. Kim, "A heuristic search-based motion correspondence algorithm using fuzzy clustering", *International Journal of Control, Automation and Systems,* vol. 10, no. 3, pp. 594-602, 2012.
[20] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, D. Estrin, "Habitat monitoring with sensor networks", Communications of the ACM 47 (6) (2004) 34-40.
[21] XIE, Xiaohui, et al. "Method and apparatus for identifying a gesture based upon fusion of multiple sensor signals." WIPO Patent No. 2013082806. 14 Jun. 2013.
[22] Gold, Ben, Nelson Morgan, and Dan Ellis. "Statistical Pattern Classification." Speech and Audio Signal Processing: Processing and Perception of Speech and Music, Second Edition (2011): 124-138.



Jin-Pyung Kim received his M.S. degree in College of Information and Communication Engineering from Sungkyunkwan University, Suwon, Korea, in 2006. He is currently a Ph.D. candidate in Sungkyunkwan University, Suwon, Korea. His research interests include structure learning, pattern recognition, and computer vision.

Gyu-Jin Jang received his M.S. degree in College of Information and Communication Engineering from Sungkyunkwan University, Suwon, Korea, in 2011. He is currently a Ph.D. candidate in Sungkyunkwan University, Suwon, Korea. His research interests include artificial intelligence, computer vision, and pattern recognition.

Jae-Young Jung received his M.S. and Ph.D. degrees in College of Information and Communication Engineering from Sungkyunkwan University, Suwon, Korea, in 1993 and 1997, respectively. He is currently an Associate Professor in Computer and Information Warfare at Dongyang University. His research interests include computer vision, pattern recognition, and image compression.

Moon-Hyun Kim received the B.S. degree in Electronic Engineering from Seoul National University in 1978, the M.S. degree in Electrical Engineering from KAIST, Korea, in 1980, and the Ph.D. degree in Computer Engineering from the University of Southern California in 1988. From 1980 to 1983, he was a Research Engineer at the Daewoo Heavy Industries Co., Seoul. He joined the College of Information and Communication Engineering, Sungkyunkwan University, Seoul, Korea in 1988, where he is currently a Professor. In 1995, he was a Visiting Scientist at the IBM Almaden Research Center, San Jose, California. In 1997, he was a Visiting Professor at the Signal Processing Laboratory of Princeton University, Princeton, New Jersey. His research interests include artificial intelligence, image recognition, and machine learning.