An Autonomic Sensing Framework for Body Sensor Networks

Surapa Thiemjarus

Imperial College/Department of Computing,
180 Queens Gate, SW7 2AZ, London, England
st01@doc.ic.ac.uk

Guang-Zhong Yang

Imperial College/Department of Computing,
180 Queens Gate, SW7 2AZ, London, England
gzy@doc.ic.ac.uk

Abstract

This paper presents an autonomic sensing framework for distributed inferencing, which consists of several self-contained machine learning techniques. A multi-objective *Bayesian framework for feature selection* is used for learning the relationship of the variables. To cater for fault tolerance and minimal resource utilisation, feature redundancy and network complexity measures have been introduced. We demonstrate how factor graphs and the sum-product algorithm can be used for model representation and inferencing. We will also show how they can be used to facilitate the mapping of model architecture onto the physical sensor networks.

Categories and Subject Descriptors I.5.2 [**Pattern Recognition**]: Design Methodology – classifier design and evaluation, feature evaluation and selection, pattern analysis; C.3 [**Special-Purpose and Application-Based System**]: Signal processing systems

General Terms Algorithms, Design, Experimentation.

Keywords multi-objective feature selection, factor graphs, pervasive monitoring, context sensing, activity recognition

1. Introduction

The recent development of wireless *Body Sensor Networks* (BSNs) has provided a basis for pervasive monitoring of patients as they go about their daily activities. The realisation of autonomic sensing is important in that it is important to large scale deployment of BSNs. The term *Autonomic Sensing* follows the concept of *Autonomic Computing* initiated by IBM in 2001 to address issues related to the rapidly growing complexity of computer systems and devices. As stated by Kephart and Chess in 'The Vision of Autonomic Computing' [1], the essence of autonomic computing is to develop self-management systems that are free from complicated system administration tasks. While the definition of autonomic sensing is likely to evolve as the contributing technologies mature and become more established, the eight *self-** properties, *i.e.* self-management, self-configuration, self-optimisation, self-healing, self-protection, self-adaptation, self-integration, and self-scaling, have highlighted some of the major requirements and challenges faced by the pervasive computing community [2].

In pervasive monitoring applications, the number of sensors can grow rapidly as they become ubiquitous. A centralised approach for sensor information processing in this case becomes impractical. Existing research has shown that the graphical probabilistic representation and the message-passing inferencing mechanism are potentially useful for dealing with uncertainties in distributed sensing systems. Paskin and Guestrin [3] proposed a robust message passing algorithm for reasoning in a *junction tree* model and verified their algorithm with a distributed sensor calibration task. By converting a standard multiply connected *Bayesian Network* (BN) into a cluster tree and combining nodes into a clique, the problem of non-convergence and incorrect update of the posterior probabilities due to loopy feedback in a multiply connected model can be avoided. The disadvantage of the junction tree algorithm is that the number of messages for communication is exponential to the size of the clique. Nevertheless, empirical studies [4, 5] have shown that *Loopy Belief Propagation* (LBP) generally provides a good approximation of the correct beliefs and the convergence is schedule invariant in a Gaussian model. In some rare cases, the algorithm does not converge. Crick and Pfeffer [6] have illustrated that LBP is a suitable means of communicating beliefs in sensor networks because of its compact implementation and distributed nature.

In this paper, a distributed inferencing framework based on the graphical probabilistic representation is proposed for pervasive monitoring with BSNs. The current framework design focuses on the following considerations: *self-optimisation* (utilisation of hardware and communication resources), *self-healing* (ability to derive decisions when data is noisy or missing), *self-adaptation* (model evolution as the mobile BSN moves through the static ambient sensor fields), and *self-scaling* (ability to deal with introduction of decision tasks and sensor nodes).

In the next section, the basic concepts of the design of the distributed decision model will be discussed. Section 3 describes a multi-objective feature selection algorithm used to associate relevant features to each activity in a distributed model. In Section 4, the design of the distributed inferencing framework will be introduced. The framework consists of two parts: model learning and model representation. In Section 5, experimental validation results of the proposed method will be presented.

2. Basic Concepts

In a pervasive monitoring application, decisions about a user can be made from the observed sensory signals from BSNs and ambient sensors. Some sensors can be more informative with regard to certain activities than others. For example, in a BSN, the motion

SIGPLAN'05 June 12–15, 2005, Location, State, Country.

signals from sensors placed on the lower limbs can provide more direct information about activities such as sitting, standing and walking, whereas sensors placed on the upper limbs are more appropriate for detecting activities such as typing and hand shaking. Ambient sensors, on the other hand, are informative only when they are in the range of the user. As a user moves through static ambient sensors in a home monitoring environment, for example, the availability of sensors to the inferencing model changes. Reasoning about different activities, therefore, will require different sets of sensors at different locations and time. When some sensors associated to a decision are unavailable, the decision can only be made based on the partial information. When all sensors associated with a decision are unavailable, only the default state of the decision can be assumed. For example, in detecting 'cooking' activity, sensors in the kitchen are more informative. When the user is not in the kitchen, the information from these sensors becomes less useful, and we can infer from the absence of information from the kitchen sensors that the user is not cooking. These scenarios show that only the information from a subset of context sensors is required for decision making at a given time.

It is known that the number of classes involved in a pervasive monitoring environment can be large depending on the granularity of the context to be captured. Instead of using a single complex model, it is possible to construct a separate model for each class. For example, a common practice in speech recognition is to represent each class (*e.g.* word or phoneme) by a *Hidden Markov Model* (HMM). This allows the intrinsic dynamics of each class to be learned separately. Classification can be made by comparing the likelihood of each HMM generates the observed sequence. In this way, a higher classification accuracy can be achieved. However, storing and inferencing with a set of HMMs can be computationally expensive.

In our previous work, we have developed a distributed inferencing framework in a similar manner, but in a way that the computational resources can be minimised. It has been shown by Tom Mitchell [7] that a naïve Bayesian classifier can be transformed into an equivalent hierarchical naïve Bayesian classifier under appropriate assumptions. Therefore, a large decision node can be decomposed into a set of smaller decision nodes, each of which represents the likelihood of the presence of a particular class. Provided that the same feature set/observation vector is used, the decomposition does not affect the posterior distribution of each class. However, since some features are more informative for a certain class than others, those features that are irrelevant to the decision can be eliminated. As a result, the inferencing comprising a set of smaller classifiers, each of which can be learned based on different sets of features can be obtained.

This type of model is particularly suitable for pervasive monitoring applications for several reasons. First, depending on the context, sensors that are irrelevant to the decision making can be put into an inactive mode to save power consumption (*i.e.* turning off the wireless channel and put the processor into the sleeping mode). Second, less computation is required when only a subset of decisions and features need to be considered for a given context. Third, decomposing a complex decision into a set of simpler decisions allows them to be computed in a distributed (bottom-up) manner. Finally, the model is more expressive as it allows the detection of a combination of context. For example, a combination of activities like ('sitting' and 'typing') or ('standing' and 'hand shaking') can be detected at the same time. Furthermore, constructing a new decision model and updating of an existing decision model can be done independently to each other. When a new activity class or more sensors are introduced, the model can be incrementally updated. At a higher level, the decision models can be connected so that the belief in one decision can also influence the belief of others. For example, they can be mutually exclusive (*e.g.* 'sitting' *vs* 'standing'), enhancing the likelihood of each other (*e.g.* 'cooking' and 'in the kitchen') or independent in nature (*e.g.* 'punching' *vs* 'running').

3. A Multi-Objective Feature Selection Algorithm

The basic structure of the proposed framework relies on a multi-objective feature selection algorithm. Feature selection is a common machine-learning technique for reducing computational complexity while maintaining the inferencing accuracy. The technique plays an important role in model construction for identifying features that are relevant to each activity.

In our previous studies, we have proposed the use of a *Bayesian Framework for Feature Selection* (BFFS), a filter-based algorithm derived from the Bayesian theory and the *Receiver Operating Characteristic* (ROC) curve, for optimal selection of sensor locations. The experimental results in these studies [8-10] indicate that the algorithm is efficient in eliminating irrelevant and redundant features. In BSN applications, however, a good feature also depends on factors such as the quality and availability of sensors, as well as communication cost among sensor nodes. In this section, we will extend the backward BFFS algorithm with a multi-objective evaluation function. To address the self-healing and self-optimisation properties of the proposed framework, two new objective measures will be introduced, *i.e.* feature redundancy and model complexity.

3.1 The Backward Multi-Objective BFFS Algorithm

In BFFS, a feature f_d which minimises the objective function $\mathrm{D}(f_i)$ will be eliminated from the feature set $\mathcal{G}^{(k)}$ in each backward feature selection step to give a selected feature subset $\mathcal{G}^{(k)} - \{f_d\}$. To maximise the model performance in terms of the aggregated discriminability power, the original BFFS prefers the feature set which maximises the expected *Area Under the ROC Curve* (AUC). This is equivalent to discarding, at each step, the feature that its presence contributes to the smallest change in the expected AUC.

$$\Delta(f_i) = E_{AUC}\left(\mathcal{G}^{(k)}\right) - E_{AUC}\left(\mathcal{G}^{(k)} - \{f_i\}\right) \tag{1}$$

where $\mathcal{G}^{(k)} = \{f_i,\ 1 \le i \le k\}$ denotes the feature set at the beginning of the iteration k, and $E_{AUC}()$ is a function that returns the expected AUC given by its parameter. Since the discriminability of the feature set before feature elimination, $E_{AUC}\left(\mathcal{G}^{(k)}\right)$, is constant regardless of f_i, omitting the term therefore does not affect the ranking of the features.

Similar to the formulation used in BFFS, feature selection can be regarded as a search problem but now with a multi-objective evaluation function. In each step of backward elimination, a feature which minimises the objective function, *i.e.* $f_d = \arg\min_i \mathrm{D}(f_i)$, will be eliminated from the feature set. A common method for formulating a multi-objective evaluation function is to use a weighted composite function as follows:

$$\mathrm{D}(f_i) = \left(1 - \sum_{j}^{M-1} \omega_j\right) O_1(f_i) + \sum_{j}^{M-1} \omega_j O_{j+1}(f_i) \tag{2}$$

where M is the number of objective measures, O_j is the j^{th} objective measure associated to f_i and ω_j is a weighting factor ranging between 0 and 1. The corresponding algorithm design for the multi-objective BFFS method is described in Table 1. The evaluation function $\mathrm{D}(f_i)$ can be replaced by one of the evaluation functions defined in Eqs. (1), (3) and (5).

Table 1. Pseudo code for multi-objective BFFS backward elimination algorithm.

(a) Let $\mathcal{G}^{(k)}$ be the full feature set and k be the size of the full feature set;

(b) Calculate the discriminability differential matrix $\mathrm{M}(f_i, f_j)$

$$\mathrm{M}(f_i, f_j) = E_{AUC}(\{f_i, f_j\}) - E_{AUC}(\{f_j\})$$

for $f_i \in \mathcal{G}^{(k)}, f_j \in \mathcal{G}^{(k)}$ and $f_i \neq f_j$;

(c) If $k = K$, output $\mathcal{G}^{(k)}$;

(d) For $f_i \in \mathcal{G}^{(k)}$ $(i = 1 \cdots k)$:

- Select k_s features from $\mathcal{G}^{(k)}$ to construct a feature subset $\mathcal{H}^{(k_s)}$. The criterion of the selection is to find k_s features f_j, for which $\mathrm{M}(f_i, f_j)$ is the smallest, where $f_j \in \mathcal{G}^{(k)}$ and $f_i \neq f_j$;
- Calculate $\mathrm{D}(f_i)$;

(e) Select feature f_d with the smallest $\mathrm{D}(f_i)$ and set $\mathcal{G}^{(k)} = \mathcal{G}^{(k)} - \{f_d\}$;

(f) $k = k - 1$; go to (c).

3.2 A Measure of Feature Redundancy

Under the proposed framework, while irrelevant features are uninformative, redundant features can be useful features although their presence may not necessarily increase the discriminability power of the feature set. The original BFFS algorithm, however, treats irrelevant and redundant features in a similar manner since both types of features contribute little to the overall model performance. The order of feature elimination, in this case, is not taken into account as long as the discriminability power of the selected feature set, $E_{AUC}(\mathcal{G}^{(k)} - \{f_i\})$, is maintained. To enhance the robustness of the system, however, redundant features can be used to improve the reliability and fault tolerance of the model. The irrelevant features should therefore be removed first during feature elimination.

As an example, Figure 1 depicts the discriminability of some features in the k^{th} iteration of BFFS. Suppose $\Delta(f_b)$ is equal to $\Delta(f_c)$ and there exists no feature interaction among features. Since features f_b and f_c provide the same additional discriminability, a higher value of $E_{AUC}(f_c)$ indicates that there is a higher overlap between the discriminability of f_c and the feature set $\mathcal{G}^{(k)} - \{f_c\}$ than that of f_b. This overlap area can be used as a measure of feature redundancy. From this diagram, the original BFFS will eliminate f_a as it provides no additional discriminability to the feature set. However when feature redundancy is taken into account, f_d should first be eliminated. This is because although the contribution of f_a is small, the discriminability of f_d on its own is minimum. The high value of $E_{AUC}(f_i)$ indicates that the feature is informative and can be used to distinguish a redundant feature from an irrelevant feature.

A new objective function for maximising the discriminability of the selected feature set while minimising the discriminability of the eliminated feature can therefore be formulated as follows:

$$\mathrm{D}_r(f_i) = -(1-\omega_1) \times E_{AUC}(\mathcal{G}^{(k)} - \{f_i\}) + \omega_1 \times E_{AUC}(f_i) \quad (3)$$

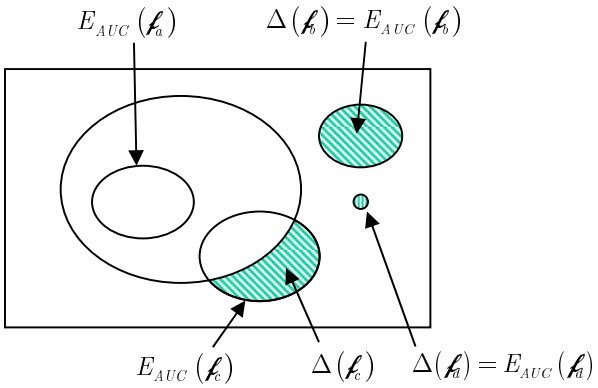

Figure 1. A schematic illustration of the discriminability of features. $\Delta(f_i)$ denotes the additional discriminability f_i adds to the features set $\mathcal{G}^{(k)}$, and $E_{AUC}(f_i)$ denotes the discriminability of f_i.

where ω_1 is the weighting factor ranging between 0 and 1. It can be shown that $\omega_1 = 1/n$ indicates that the additional discriminability is n times more important than the overlap discriminability. When $\omega_1 = 0$, this is equivalent to the original BFFS objective function.

3.3 Estimating the Implementation Cost

For BSNs, the model complexity in terms of implementation cost should be minimised whilst maximising the feature discriminability and fault tolerance (through feature redundancy). At the lowest level, a large number of features implies higher model complexity and computational resources. Features from different sensors introduce a higher hardware cost than features derived from the same sensor as this reduces the number of sensor nodes used and the wireless communication cost. In BSN, network communication usually dominates the power consumption [11] and the formation of a subnet with message hopping is a unique feature of low-power short range radio used for wireless sensor networks [12]. Features derived from sensors on the same node can be directly fused and therefore will incur no communication cost. Since a BSN is mobile, the interactions of the BSN with the surrounding WSN can be highly dynamic. The connection between two subnets may need to be established and authenticated at run-time; therefore, fusing features from different subnets can introduce a higher cost than fusing features from the same subnet. The cost of features used for making a decision can be estimated based on the number of features, the number of sensors used and different levels of communication incurred: 1) features on the same sensor node; 2) features from different sensor nodes in the same subnet; and 3) features from different subnets (*e.g.* between a BSN and ambient sensors).

Let $count\_subnet()$ be a function that returns the number of subnets and $count\_sensor()$ a function that returns the number of sensors from which the features in the argument are obtained. The estimation of the communication cost can be defined as:

$$C = \left(count\_subnet\left(\mathcal{G}^{(k)}\right)\right)^2 \times count\_sensor\left(\mathcal{G}^{(k)}\right) \quad (4)$$

One of the issues related to the above weighted composite evaluation function is that the solution is dependent on the underlying weighting vector and the range of each objective function. To alleviate this problem, each factor involved in the above function is expressed in this study in a normalised form. The evalua-

tion function for optimising the model performance and the communication cost can therefore be formulated as:

$$\mathrm{D}_c(\mathcal{F}) = (1-\omega_1-\omega_2)\times\frac{-E_{AUC}\left(\mathcal{G}^{(k)}-\{\mathcal{F}\}\right)}{E_{AUC}\left(\mathcal{G}^{(k)}\right)}+\omega_1\times\frac{E_{AUC}(\mathcal{F})}{E_{AUC}\left(\mathcal{G}^{(k)}\right)} \quad (5)$$
$$+\omega_2\times\left(\left(\frac{count\_node\left(\mathcal{G}^{(k)}-\{\mathcal{F}\}\right)}{count\_node\left(\mathcal{G}^{(k)}\right)}\right)^2\times\frac{count\_sensor\left(\mathcal{G}^{(k)}-\{\mathcal{F}\}\right)}{count\_sensor\left(\mathcal{G}^{(k)}\right)}\right)$$

where ω_2 is a relative weighting factor.

4. A Distributed Inferencing Framework

From the above feature selection algorithm, we will focus in this section on the actual design of the distributed inferencing framework. Based on the information architecture proposed by Chu *et al*. [13], design issues for the development of a distributed inference algorithm in *ad hoc* sensor networks can be addressed. Our emphasis is on model representation and evolution issues. By separating modelling from the representation layer, modifications on each layer can be made without affecting the other.

4.1 Model Learning

In terms of model learning, there are two main approaches for structure learning for a probabilistic model. The dependency analysis approach aims to uncover the dependencies from the data using a statistical test on hypotheses. The search-and-scoring based approach formulates the structure learning as an optimisation problem of finding the model structure which maximises the score. In this study, we propose a hybrid approach for model learning.

To simplify the structure search problem, the dependency graph is constrained so that only direct dependency between a decision variable, A_j, and a feature variable, F_i, is allowed. An example of an initially fully connected dependency graph which conforms to this model description is shown in Figure 2. The backward BFFS algorithm described in the previous section provides a systematic way of determining feature variables that should be included in the inferencing model and their relationship with each decision. With backward feature selection, the dependency graph construction can be regarded as the elimination of weak dependency links associated with each decision node one by one. The association with features can be learned independently for each decision. In terms of scalability, this indicates that a new decision node and more features can be incrementally introduced without affecting the accuracy of existing decisions. Updates on features associated to an existing decision node can also be made without affecting others.

After the dependency structure is obtained, the next step is to assign a causal direction to each link. This can be achieved through feature analysis, *i.e.* a multiple parent configuration is required when there exists either some correlation or some interaction among the observable features. However, a single parent structure is assumed in this paper for the sake of simplicity. The dependency and causality information determines how the joint probability of the variables in the model can be factorised. For model parameter learning, co-occurrence matrices of variables involved in each local conditional distribution are first generated. Each matrix can then be converted into a probability distribution by dividing each element in the matrix by the number of data points. A small constant can be added to each element of the matrices prior to normalisation so as to enhance model generalisability to unseen cases.

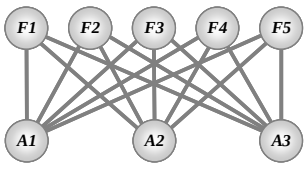

Figure 2. An example of the initial dependency graph.

4.2 Model Representation, Inference and Deployment

As a user moves through a network of ambient sensors, the availability of sensors in the vicinity of the BSN worn by the user should be updated. This information can be used to determine a subset of possible decisions that could be inferred. Thus, only certain segments of the dependency graph are needed to be activated at a given time. This can be viewed as online evolution (or self-adaptation) of the representation as the BSN entering, moving though, and leaving ambient sensor networks. The suitable deployment of the distributed model in the physical network varies, depending on the actual hardware implementation. Two examples of possible schemes are:

- *All decisions are made on the BSN node*: In this scheme, each BSN node maintains a copy of the whole model. Dynamic connection can be established for transmission of the raw or pre-processed data from the activated ambient sensors to the BSN node for reasoning about the context of its owner. Only a subset of features is instantiated as the user moves through the sensor field. The inferred contexts of the user can be stored on the BSN node, or transmitted to the database via the *Wireless Sensor Network* (WSN) infrastructure as necessary;

- *Decisions are made on ambient sensor nodes when possible*: Fusion of decisions that involve only features from on-board sensors can be made on the BSN node and only the results are sent out to the database. Fusion of decisions that involve features from ambient sensors can be made on an ambient node. The BSN node can broadcast its ID and relevant features for decision fusion. The ID can be used for identifying the user to which the inferred context belongs. The inferred information can be collected and transmitted to the database via a local area network. In this scheme, only simplex data transmission, *i.e.* from a BSN to the ambient nodes, is required. The model can be spitted into segments and distributedly embedded throughout the physical sensor network. Only minimally compact representation will be kept on each BSN node. Different users can share the representation stored on the ambient nodes.

While the first scheme is simple, the second one is more practical in a large network. In this scheme, optimal mapping of the logical model onto the actual physical network becomes an important issue. To facilitate this, we propose the use of *Factor Graph* (FG) for model representation. A FG is a bipartite graph that represents the global function $g(x_1,\ldots,x_n)$ as a product of J local functions $f_j(X_j)$ that map a subset X_j of $\{x_1,\ldots,x_n\}$ to some range. In general, a FG consists of two types of nodes: variable nodes and function (or factor) nodes. The presence of an arc from a variable node to a factor node indicates that the variable is an argument of the function.

A FG can be used to describe general functions other than probability distributions (*e.g.* logical or logistic functions), and thus have more expression power than a BN. Recall that the distribution represented by a BN can be written as:

$$P(x_1,\ldots,x_n)=\prod_{i=1}^{n}P(x_i\mid pa_i) \quad (6)$$

where pa_i denotes the set of parents of x_i. According to [14], a BN can be translated into a FG simply by introducing a function node for each factor $P(x_i \mid pa_i)$ and drawing edges from this factor node to x_i and its parents pa_i.

In cases where a feature node is shared among several decision nodes, the link matrix associated to the node can be very large and impractical. Unlike BN, FG representation allows certain simplified assumptions to be asserted so that the probabilistic model is more compact and suitable for the deployment in a WSN. For example, let feature $F3$ is shared between decision nodes $A1$ and $A3$. By asserting the independency assumption between the two decisions, the conditional probability $P(F3 \mid A1, A3)$ can be simplified as:

$$\begin{aligned} P(F3 \mid A1, A3) &= \frac{P(A1, A2 \mid F3)P(F3)}{P(A1, A2)} \\ &= \frac{P(A1 \mid F3)P(A2 \mid F3)P(F3)}{P(A1)P(A2)} \\ &= \frac{P(F3 \mid A1)P(A1)}{P(F3)} \frac{P(F3 \mid A2)P(A2)}{P(F3)} \frac{P(F3)}{P(A1)P(A2)} \\ &= \alpha P(F3 \mid A1)P(F3 \mid A2) \end{aligned}$$

where $\alpha = 1/P(F3)$ is the normalisation constant and can be omit from the model.

For model inference, the sum-product algorithm [14] provides an efficient way to compute all the marginal functions in the factor graph. The algorithm operates according to the following simple rule:

> *"The message sent from a node v on an edge e is the product of the local function at v (or the unit function if v is a variable node) with all messages received at v on edges other than e, summarised for the variable associated with e"*

In a cycle-free FG, a schedule in which only pending messages are transmitted will yield exact function summaries when the sum-product algorithm terminates with no further messages pending. In a FG with cycles, however, such a state condition cannot be reached, since the transmission of a message will recurrently trigger a calculation of another message in the cycle. One way to develop a finite schedule is to alternatively propagate messages between all variable nodes and all factor nodes for a specified number of iterations. A LBP inferencing algorithm can be adopted based on this message-passing schedule. This introduces more compact link matrices compared to the node clustering technique. It also ensures that message passing can be made in parallel without waiting for messages from other nodes. Table 2 describes the inference algorithm used in the proposed FG model. In our implementation of the FG inference engine, the non-root variable nodes are associated with an extra factor node with default uniform state values (*i.e.* all 1s). During variable instantiation, the evidence for each feature node enters the network by updating the values of the local function associated to the node.

In the FG representation, variable nodes are used to represent the states and measurements, while function nodes are used to represent the models. The separation in the representation of the actual measurements and the function storage facilitates the mappings of the logical model onto the physical network. According to [13], the optimal node assignment $\hat{A}$ can be made in such a way that the cost of communication is minimised, *e.g.*

Table 2. The inference algorithm for the proposed FG model.

(a) Initialise all elements of the outgoing messages to 1s;

(b) Enter the evidence for each feature (instantiated node);

(c) For iteration 1 to n :

- Update all outgoing messages from each variable node *x*:
$$\mu_{z \to f(z)} = \prod_{h \in n(z) \setminus \{f\}} \mu_{h \to z}(x)$$
- Pass the messages to the corresponding neighbour factor nodes;
- Update all outgoing messages from each factor node f :
$$\mu_{f(z) \to z}(x) = \sum_{\sim \{z\}} \left(f(x) \prod_{y \in n(f) \setminus \{z\}} \mu_{y \to f}(y) \right)$$
- Pass the messages to the corresponding neighbour variable nodes;

(d) Calculate the posterior probability distribution of each decision A by normalising the product of all the incoming messages of node A.

$$\hat{A} = \arg_{A \in \mathbf{A}} \min \sum_{(x,f) \in E} w_{(x,f)} \cdot \text{cost}(A(x), A(f)) \quad (7)$$

where an assignment $A: V \to \{1,...,n\}$ is a mapping function from the set of all variable and function nodes in the graph V to the n sensor nodes of the sensor network, $\mathbf{A}$ is a set of all feasible assignment, $(x, f) \in E$ indicates an edge between a variable node $x \in V$ and a function node $f \in V$, $w_{(x,f)}$ is a scalar weight for the cost of sending messages along edges $(x, f) \in E$, and $\text{cost}(i, j)$ is the communication cost.

With the above model mapping scheme, the cost of model inferencing can be minimised. The multi-objective feature selection framework described earlier, on the other hand, minimises the cost of data acquisition. In some sense, the combination of the two techniques provides the proposed distributed inferencing framework with the self-optimisation property.

5. Experiments and Results

For validating the proposed framework, an exercise sensing dataset was used to demonstrate the strength of the proposed distributed inferencing method over a centralised model and to show how the multi-objective BFFS can add values to the framework. The activities of the subject included *sitting* (*chair*), *standing*, *steps*, *sitting* (*floor*), *demi-plie*, *galloping left*, *skipping*, *galloping right*, *side kick*, *front kick*, and *walking*. A seven-minute sequence was collected at the sampling rate of 30 Hz using four two-axis accelerometers placed on the left and right ankles and legs, 20% of which is used for constructing the training set and the rest is used for model validation. The dataset has evenly distributed classes. Both the raw signal and signal energy calculated over a fixed window of 50 samples (2s) for each sensor channel were used for the experiment, *i.e.* there are 16 features in total. Features 1-8 are raw signal from *right ankle* (*x*), *right ankle* (*y*), *left leg* (*x*), *left leg* (*y*), *right leg* (*x*), *right leg* (*y*), *left ankle* (*x*), and *left ankle* (*y*), respectively. Features 9-16 are the corresponding signal energy.

For this dataset, the eleven activities are mutually exclusive. To construct a distributed state model, the eleven-state decision variable of the exercise dataset was separated into multiple binary-decision variables. When each binary decision variable in the

distributed state model is connected to all features, an equivalent accuracy to a naïve BN can be achieved. As demonstrated in Figure 3, the two equivalent models yield ~77.3% classification accuracy on the unseen validation dataset.

To reduce the use of computational resources, the backward BFFS was applied for each binary decision variable in the dataset with distributed decision state. Table 3 and Table 4 show the results of the backward BFFS without and with in-built redundancy (*i.e.* $\mathrm{D}_r\left(\mathscr{f}_i\right)$ with the values of redundancy weight ω_1 equal to 0 and 0.1, respectively). In Table 3, feature selection is made based on the criterion that the AUC value is above 0.98. Without the redundancy measure, the dependency graph in Figure 4 (Model 1) is obtained. To enable a fair comparison, the same selection mask was applied to the results in Table 4. With feature redundancy, the dependency graph in Figure 5 (Model 2) is obtained. The same feature selection mask implies the same number of links in both logical models, *i.e.* comparable number of model parameters and computational operations during model inference.

From feature analysis, no multi-parent structure for each decision node was found. Based on the learned dependency graphs and causality, the FG representation of the models was constructed. With the FG inference algorithm in Table 2, Model 1 and Model 2 yield the classification accuracy of 68.28% and 72.60%, respectively. Since there exists no multi-parent structure and feature nodes are fully instantiated, the converging results can be obtained after 2 iterations. Each of the two distributed decision state models contains 34 links, compared to 176 links in the fully linked distributed decision state model. While 13 out of 16 features are included in Model 1, only 9 features are included in Model 2 when feature redundancy was taken into account.

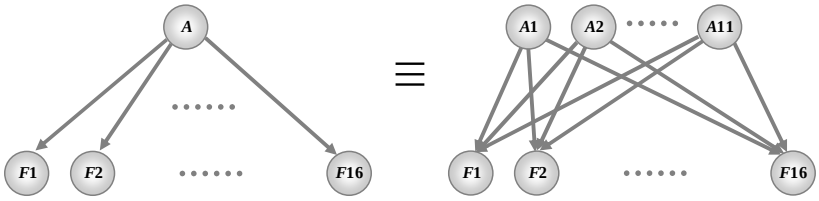

Figure 3. A naïve BN (left) and the equivalent distributed decision-state model (right) for the exercise dataset.

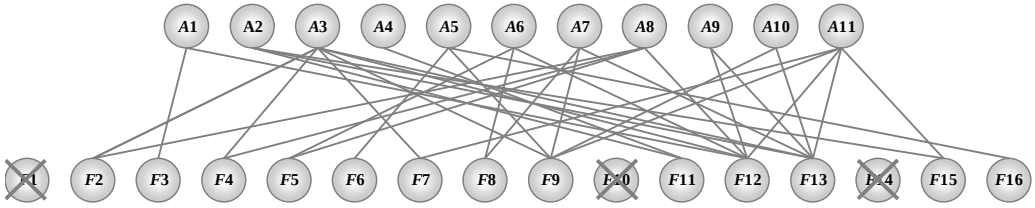

Figure 4. The dependency graph model for the exercise dataset obtained from BFFS without redundancy measure (Model 1).

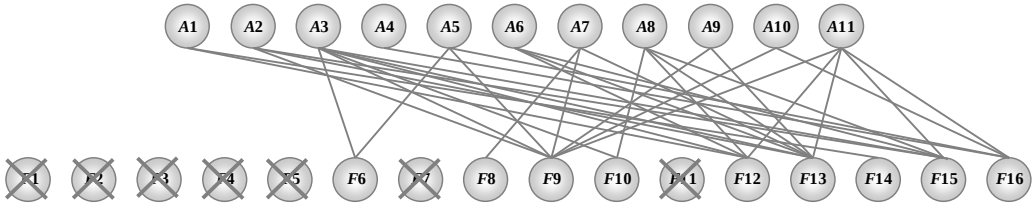

Figure 5. The dependency graph model for the exercise dataset obtained from BFFS with redundancy measure (Model 2).

Table 3. Changes of AUC in the exercise training dataset with distributed binary decision variables during BFFS backward elimination.

Sit (chair)	**AUC**	0.9725	0.9902	0.9954	0.9976	0.9987	0.9991	0.9992	0.9992	0.9992	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994	0.9994
	Feature ID	12	3	8	1	13	4	5	15	2	7	14	9	16	6	10	11
Stand	**AUC**	0.8584	0.9629	0.9806	0.99	0.9952	0.9969	0.9977	0.9981	0.9981	0.9985	0.9988	0.9988	0.9993	0.9993	0.9993	0.9993
	Feature ID	13	15	11	4	1	6	5	16	14	2	8	12	7	9	10	3
Steps	**AUC**	0.775	0.8823	0.9476	0.9622	0.9754	0.986	0.9926	0.9967	0.9989	0.9994	0.9997	0.9999	0.9999	0.9999	0.9999	0.9999
	Feature ID	13	9	12	7	4	2	6	16	15	3	5	1	14	8	10	11
Sit (floor)	**AUC**	0.9816	0.9995	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Feature ID	11	3	16	9	8	14	15	12	7	4	13	5	2	10	6	1
Demi-plie	**AUC**	0.8719	0.9575	0.9873	0.9942	0.9975	0.9991	0.9997	0.9998	0.9999	0.9999	1	1	1	1	1	1
	Feature ID	9	16	6	1	8	12	2	13	4	5	15	10	3	14	7	11
Gallop L	**AUC**	0.9414	0.9605	0.9809	0.9939	0.9991	0.9999	1	1	1	1	1	1	1	1	1	1
	Feature ID	12	5	8	7	3	6	2	1	4	9	13	10	16	15	14	11
Skip	**AUC**	0.827	0.944	0.9838	0.9972	0.9995	0.9999	0.9999	1	1	1	1	1	1	1	1	1
	Feature ID	9	13	8	1	12	7	10	16	5	2	3	4	11	6	15	14
Gallop R	**AUC**	0.8587	0.9099	0.9646	0.9907	0.9987	0.9998	1	1	1	1	1	1	1	1	1	1
	Feature ID	12	5	4	2	3	7	9	6	10	13	8	14	16	1	15	11
Side kick	**AUC**	0.9599	0.9968	0.9991	0.9997	0.9999	1	1	1	1	1	1	1	1	1	1	1
	Feature ID	13	12	1	3	5	8	2	7	15	6	10	4	14	9	16	11
Front kick	**AUC**	0.9622	0.9964	0.9997	0.9999	1	1	1	1	1	1	1	1	1	1	1	1
	Feature ID	9	13	6	7	1	2	4	16	5	8	10	14	3	12	15	11
Walking	**AUC**	0.7648	0.9203	0.9446	0.9791	0.9887	0.9941	0.9977	0.9989	0.9995	0.9998	0.9999	1	1	1	1	1
	Feature ID	9	12	13	15	7	8	1	16	11	4	6	2	10	3	5	14

Table 4. Changes of AUC in the exercise training dataset with distributed binary decision variables during mutli-objective BFFS backward elimination with redundancy weight ω_1 equals 0.1.

Sit (chair)	**AUC**	0.9809	0.99	0.99	0.9925	0.9931	0.997	0.9977	0.9977	0.9978	0.9983	0.9991	0.9991	0.9991	0.9991	0.9992	0.9994
	Feature ID	14	12	11	6	3	1	4	10	9	8	13	2	16	15	7	5
Stand	**AUC**	0.8584	0.9629	0.9787	0.989	0.9908	0.9933	0.9946	0.9966	0.9972	0.9972	0.9974	0.9974	0.9974	0.9979	0.9991	0.9993
	Feature ID	13	15	9	4	10	5	12	8	6	16	2	14	11	3	1	7
Steps	**AUC**	0.7827	0.9154	0.9436	0.968	0.9762	0.9816	0.9912	0.992	0.9967	0.9984	0.9991	0.9993	0.9996	0.9997	0.9997	0.9999
	Feature ID	15	13	12	9	6	10	7	14	4	16	5	3	1	8	11	2
Sit (floor)	**AUC**	0.9852	0.9993	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	Feature ID	16	3	11	9	10	6	15	14	1	8	7	13	2	12	5	4
Demi-plie	**AUC**	0.8719	0.9575	0.9873	0.9937	0.9969	0.9981	0.9988	0.9993	0.9997	0.9998	0.9999	0.9999	1	1	1	1
	Feature ID	9	16	6	12	1	15	13	10	3	5	2	8	7	4	14	11
Gallop L	**AUC**	0.9414	0.9613	0.9683	0.982	0.9888	0.9943	0.9987	0.9991	0.9991	0.9999	1	1	1	1	1	1
	Feature ID	12	13	15	9	10	4	2	16	14	8	5	7	3	1	6	11
Skip	**AUC**	0.827	0.944	0.9838	0.9884	0.9961	0.9993	0.9999	1	1	1	1	1	1	1	1	1
	Feature ID	9	13	8	15	3	12	1	6	5	4	10	16	2	11	14	7
Gallop R	**AUC**	0.8597	0.9266	0.9571	0.9644	0.9819	0.995	0.9962	0.9995	0.9996	1	1	1	1	1	1	1
	Feature ID	10	12	13	15	4	5	14	7	16	8	9	3	2	6	1	11
Side kick	**AUC**	0.9599	0.9863	0.9916	0.9937	0.997	0.9991	0.9991	1	1	1	1	1	1	1	1	1
	Feature ID	13	9	10	14	3	12	15	1	5	11	2	7	8	6	16	4
Front kick	**AUC**	0.9622	0.9828	0.9972	0.9975	0.9978	0.9982	0.9994	0.9998	1	1	1	1	1	1	1	1
	Feature ID	9	16	10	15	12	3	13	2	1	4	8	7	14	11	6	5
Walking	**AUC**	0.7648	0.9203	0.9446	0.9791	0.9885	0.9892	0.9961	0.9976	0.9987	0.9991	0.9998	1	1	1	1	1
	Feature ID	9	12	13	15	16	10	14	3	6	1	2	8	7	11	4	5

6. Conclusions and Discussions

In this paper, a framework for learning a distributed inferencing model for pervasive monitoring with BSNs has been proposed. The method addresses several characteristics of an autonomic system. For self-adaptation, the use of each segment of the logical model can be determined by the activation of the corresponding physical sensor nodes. In other words, the model adapts/evolves itself according to the availability of information from sensors in the BSN vicinity. The availability of the information is in turn affected by the change in user context (*e.g.* location). The multi-objective BFFS algorithm provides a systematic method for selection of relevant sensory channels for model construction. The technique is self-contained and can also be applied to other applications. To achieve self-healing and fault tolerance, the proposed method integrates the redundancy measure into the evaluation function. In addition, when some features associated with a decision node is missing, the feature nodes are simply un-instantiated and the reasoning can still be made based on the existing features. For self-optimisation, model complexity during data acquisition is addressed by the use of feature selection and the implementation cost measure. Cost minimisation during model inference is addressed by the logical-physical model mapping schemes. To achieve self-scaling, the framework can seamlessly adjust with added system resources and information processing tasks, *i.e.* a new decision node and more features can be incrementally introduced and updates on features associated to an existing decision node can be made without affecting the others.

By separating model learning from the representation layer in the framework design, modifications can be made in one layer without explicit involvement of the others, *e.g.* changes in the prior constraints on general model structure for better modelling accuracy are transparent of the representation layer, and changes in the logical-physical mapping scheme or the inference engine can be made without affecting how the model is learned.

Initial experiment results have shown that with a distributed model, significant resource savings and a reasonable accuracy can be achieved. With a similar dependency structure, the model with feature redundancy can yield a higher classification accuracy. Although the accuracy of the learned distributed model presented this paper is slightly lower than the centralised naïve BN, further improvement in accuracy can be achieved by modifying the feature selection masks to allow more features to be used for decision classes with low accuracy.

References

[1] Kephart, J.O. and Chess, D.M. The vision of autonomic computing IEEE Computer Magazine, January 2003, 41-50.

[2] Yang, G.-Z., Lo, B.P.L. and Thiemjarus, S. Autonomic sensing. In Yang, G.-Z. ed. Body Sensor Networks, Springer-Verlag, London, 2006, 333-372.

[3] Paskin, M.A. and Guestrain, C.E., Robust probabilistic inference in distributed system. In the Twentieth Conference on Uncertainty in Artificial Intelligence, (Banff, Canada, 2004).

[4] Murphy, K., Weiss, Y. and Jordan, M.I. Loopy belief propagation for approximate inference: an empirical study the Fifteenth Conference on Uncertainty in Artificial Intelligence, 1999, 467-475.

[5] Weiss, Y. and Freeman, W.T. Correctness of belief propagation in gaussian graphical models of arbitrary topology. Neural Computation, 13 (10). 2173-2200.

[6] Crick, C. and Pfeffer, A., Loopy belief propagation as a basis for communication networks. In the Nineteenth Conference on Uncertainty in AI, (Acapulco, Mexico, 2003).

[7] Mitchell, T. Conditions for the equivalence of hierarchical and non-hierarchical bayesian classifiers, Center for Automated Learning and Discovery, Carnegie Mellon University, Pittsburgh, PA, 1998.

[8] Thiemjarus, S., Lo, B.P.L. and Yang, G.Z., Feature selection for wireless sensor networks. In the First International Workshop on Wearable and Implantable Body Sensor Networks, (Imperial College, London, 2004), IEE.

[9] Yang, G.-Z. and Hu, X.P. Multi-Sensor Fusion. In Yang, G.-Z. ed. Body Sensor Networks, Springer-Verlag, London, 2006, 239-286.

[10] Hu, X.P. Feature selection and extraction of visual search strategies with eye tracking, Imperial College, 2005.

[11] Raghunathan, V., Schurgers, C., Park, S. and Srivastava, M.B. Energy-aware wireless microsensor networks. IEEE Signal Processing Magazine, 19 (2).

[12] Yang, G.-Z. (ed.), Body Sensor Networks. Springer-Verlag, London, 2006.

[13] Chu, M., Mitter, S.K. and Zhao, F., An information architecture for distributed inference on ad hoc sensor networks. In the Forty-First Annual Allerton Conference on Communication, Control, and Computing, (Monticello; IL; USA, 2003).

[14] Kschischang, F., Frey, B.J. and Loeliger, H.-A. Factor graphs and the sum-product algorithm. IEEE Transactions on Information Theory, 47 (2).