DYNAMIC FACTOR GRAPHS: A NOVEL FRAMEWORK FOR MULTIPLE FEATURES DATA FUSION

Kittipat Kampa, Jose C. Principe and K. Clint Slatton

Department of Electrical and Computer Engineering, University of Florida, Gainesville, FL

ABSTRACT

The Dynamic Tree [1] (DT) Bayesian Network is a powerful analytical tool for image segmentation and object segmentation tasks. Its hierarchical property makes it possible to analyze and incorporate information from different scale which is desirable in many applications. Having flexible structure enables model selection capability, which is much desired recently, occur at the same time in the process. In this paper, we propose a novel framework, dynamic factor graphs (DFG), where data segmentation and fusion are combined in the same framework. The new framework is built on 2 main ideas; 1) flexible structure (inherited from the dynamic tree) and 2) factor graphs (FGs). FGs enable us to have a broader range of modeling application than Bayesian networks (BNs) since FGs include both directed acyclic and undirected graphs in the same setting. The example in this paper will focus on 2D multiple features data segmentation and fusion with linear Gaussian model assumption.

Index Terms— dynamic factor graphs, sum-product algorithm, linear Gaussian models, data fusion, data segmentation

1. INTRODUCTION

Dynamic tree (DT) [1] was originally invented in order to model images in a probabilistic and hierarchical manner. The DT is developed from the earlier work on Tree-Structured Belief Network (TSBN) [2] which provides a natural way of modeling image within a probabilistic framework. TSBN has a desirable hierarchical structure, but suffers from “blocky” segmentation due to its fixed structure. Therefore, DT was invented to solve this problem by allowing the structure of the network to change in order to fit the underlying distribution. Having a flexible structure, DT can generally fit a given distribution better, and it shows promise to eliminate the blocky segmentation that occurs with TSBN. A common algorithm to infer a distribution at a node in both TSBN and DT is Pearl’s message passing [3], which gives an attractive linear-time algorithm when the directed graph structure does not contain any loop.

Bayesian networks (BNs) and Markov random fields (MRF) are classes of probabilistic graphical model corresponding to directed acyclic graphs (DAG) and undirected graphs (UG) respectively. Both DT and TSBN were invented based on BN, so both are limited to only DAG. In order to incorporate both DAG and UG together in the same framework, the factor graph (FG) [4] was invented together with its time-efficient inference algorithm sum-product algorithm [4] which is similar to Pearl’s message passing. In this paper, we propose a novel methodology that inherits the flexible structure capability of DT and FG. The new architecture is called dynamic factor graphs (DFG). DFG can incorporate both DAG and UG simultaneously, and is capable of changing structure (model selection) in the same framework. To demonstrate the power of DFG, an example application on data fusion will be illustrated.

Data fusion is an increasingly important and much needed technology due to advances in sensor technology. Moreover, there is much need for data fusion algorithms that are able to provide both the estimation and its corresponding uncertainty. This requirement is ideally suited for the probabilistic-based framework proposed in this paper. We develop a novel probabilistic framework for data fusion based on DFG, which is inherently probabilistic-based on its own accord. Additionally, the new framework is designed to incorporate multiple features and multidimensional data as well.

In the next section an overview of factor graphs and dynamic factor graph will be presented. A description of DFG for data fusion is presented next, and the DFG structure optimization is then introduced. The remaining sections explain the incorporation of extracted multiple features into the DFG framework for segmentation and sensor fusion, and results are presented last.
2. DYNAMIC FACTOR GRAPHS

In general, a factor graph (FG) is a bipartite graph that expresses how a “global” function of many variables decomposes into a product of “local” functions [4]. In this scenario, the global and local function can be thought of as the full joint distribution over all the variables in the structure and conditional distribution over some variables respectively. The conditional distribution is obtained by applying the conditional independence property of the network. The circle and the square in Fig. 1 represent a variable node and function node respectively. This framework is also as shown.

2.2. Dynamic factor graphs

A dynamic factor graph is a factor graph whose joint probability \( P(Z,V) \) includes a random variable \( Z \) that defines the network structure. The random variable \( Z \) is in the form of a matrix whose \( i^{th} \) row \( j^{th} \) column element is written as \( Z_{ij} \). Each element \( Z_{ij} \) indicates the connection between the \( i^{th} \) child node and the \( j^{th} \) parent node. \( Z_{ij} \) is 1 when such a connection is present, and 0 otherwise. The random variable \( V \) is the set of nodes in the network structure. Normally the nodes in a network can be separated into 2 types: 1) hidden nodes and 2) observed nodes. Let \( V = \{X,Y\} \) where \( X \) and \( Y \) are the set of hidden nodes and observed nodes in the network respectively. In general \( X \) and \( Y \) can be any type of random variable, but in this paper we will focus on continuous-valued random variables which are less straightforward than discrete-valued ones [5].

So far, \( Z \) describes qualitatively how variables in the network interact but does not indicate its strength. To describe such a quantity, the final parameters \textit{conditional probability distributions} (CPDs) are required. The CPD between a child node \( j \) with a parent node \( i \) is usually written in the form of \( P(V_i|V_j) \). In this paper we assume the CPD takes the form of a linear Gaussian model which will be discussed later on.

The goal of DFG is to calculate the optimal structure \( Z^* \) that can best explain the given observation \( Y \). That is we want \( Z^* = \arg\max_Z P(Z|Y) \). In terms of data fusion problem proposed in this paper, \( Z^* \) encodes a non-redundant set of objects at the root nodes and also connects each leaf node upward to its most likely root node. It is also worth mentioning that the posterior distribution of each root node captures the (fused) information obtained by combining all the information from its corresponding leaf nodes.

3. DATA FUSION FRAMEWORK

3.1. Overview of the algorithm

In this DFG framework, the data and the underlying structure are represented by a factor graph. It is usually the case that having multiple features can enhance the classification and segmentation result. This framework is designed in order to incorporate multiple features, each of which can take the form of a multidimensional feature vector. Moreover, the random variables are also assumed multivariate normally distributed. The overview of the framework is illustrated in Fig. 2.

3.2. Mathematical model of the problem

Let \( D \) denote \( \{D_1, D_2, ..., D_N\} \) a set of \( N \) acquired data. Each of the acquired datum \( D_i \) is then extracted into \( F \) multidimensional feature vectors \( D_i^{(1)}, D_i^{(2)}, ..., D_i^{(F)} \). The architecture of this framework comprises \( F \) blocks, each of which contain the same number of nodes and share the same topological structure \( Z \). Let \( X \) and \( Y \), random variables in the whole network, denote the hidden variables and observed variables respectively. Additionally, the random variable \( X \) can be separated into \( F \) independent random variables according to feature extraction. That is \( X = \{X^{(1)}, X^{(2)}, ..., X^{(F)}\} \) where \( X^{(f)} \) denotes the hidden variable \( X \) that contributes to feature \( f \in \{1, ..., F\} \). Furthermore, we can think of \( X^{(f)} \) as a set of hidden variables (nodes) contained in the block \( f \) allowing us to further factorize the variable as \( X^{(f)} = \{X_1^{(f)}, X_2^{(f)}, ..., X^{(f)}_M\} \).
where $M$ denotes number of hidden variables in each feature. The same factorization applies to $Y$, that is $Y = \{Y^{(1)}, Y^{(2)}, \ldots, Y^{(F)}\}$ and $Y^{(f)} = \{Y^{(f)}_1, Y^{(f)}_2, \ldots, Y^{(f)}_N\}$. In each block, the number of the leaf nodes is equal to the number $N$ of acquired data (e.g. images in this application). Each of the feature vector $D_i^{(f)}$ will be plugged into the corresponding leaf node $Y_i^{(f)}$, which is the $i^{th}$ leaf node in the block $f$. At this point we also say $Y_i^{(f)}$ is observed and take a vector value $D_i^{(f)}$. Again, the structure matrix is $Z = (Z_{ij})$, where $Z_{ij} = 1$ when the child node $i$ is connected to the parent node $j$ and 0 otherwise.

The main goal of DFG is to calculate the optimal structure $Z^*$. Hence we have

$$
Z^* = \arg \max_Z P(Z|Y) = \arg \max_Z (\log P(Y|Z) + \log P(Z))
$$

We assume that each feature $f$ is mutually independent so the conditional distribution can be rewritten as

$$
P(Y|Z) = \int \mathbb{P}(X,Y|Z) \, dX = \int \prod_{f=1}^{F} \mathbb{P}(X^{(f)}_1 Y^{(f)}_1, \ldots, X^{(f)}_N Y^{(f)}_N | Z) \, dX
$$

By taking $\log P(Y|Z) = \sum_{f=1}^{F} \log P(Y^{(f)}|Z)$, finally, we will have

$$
Z^* = \arg \max_Z (\sum_{f=1}^{F} \log P(Y^{(f)}|Z) + \log P(Z))
$$

Here we call the quantity inside the argmax the cost function, and we want to calculate for the structure $Z^*$ that maximizes the cost function.

In the real-world application it is more likely that each feature would be weighted differently by the human experts. Also, we want the optimal structure to be as simple as possible, which means that we would penalize more on the structure with more complexity. Consequently, we modify the original cost function to get the final cost function $C(Z)$ as follow

$$
C(Z) = \sum_{f=1}^{F} w_f \log P(Y^{(f)}|Z) + \log P(Z) + \text{pen}(Z)
$$

The first term of $C(Z)$ corresponds to the likelihood of the structure $Z$ given the evidence $Y$, and $w_f$ is the user-defined weights with the constraint $\sum_{f=1}^{F} w_f = 1$. The second term corresponds to the prior probability of the structure which can be obtained from, for example, simple segmentation. This helps us to shape the resulting structure. The last term $\text{pen}(Z)$ is the penalty term, which is a function of the structure $Z$ contributing to penalizing the complex structure. In order to calculate the optimal structure $Z^*$ such that

$$
Z^* = \arg \max_Z C(Z),
$$

which is the main goal of this paper, the sum-product algorithm is exploited to calculate the joint $P(X^{(f)}_i Y^{(f)} | Z)$, and hence we can automatically calculate the likelihood of the structure given the evidence $P(Y|Z)$ from $P(Y|Z) = \int_{X^{(f)}_i} P(X^{(f)}_i Y^{(f)} | Z) \, dX^{(f)}_i$, which is tractably computable. Recall that a structure $Z$ determines a unique factorization over the entire joint, so it would be difficult to infer a distribution at a node without sum-product algorithm. For discrete-valued $X$, we can obtain similar form by simply switching $\int_{X} \cdot$ to $\sum_{X}$.

### 3.3. DFG structure optimization

The structure $Z$ is initialized with the structure in Fig. 4 (a), and then the structure is perturbed repeatedly and progressively. The cost function $C(Z)$ is calculated for each and every structure $Z$ until the iteration stops. At the end the structure that gives maximum value of $C(Z)$ is the optimal structure $Z^*$. In general, large optimization problems can be solved successfully using stochastic search algorithm [6]. Structure optimization, as in this paper, is also considered an NP-hard problem, and simulated annealing (SA) [7] is chosen to be implemented in this work to solve the problem.

### 3.3. Interpretation of the model

DFG has hierarchical properties where the information can be visualized in different scale, and it is usually the case that
interpretation involving generative model of the data can be made. In this sensor fusion application illustrated by Fig. 4, we have 3 levels to consider: 1) root nodes represent the estimate of the true feature, 2) middle-layer nodes represent sensor platform ID who acquire the data and there are 2 platforms in this example, and 3) leaf nodes represent the realization of the true feature.

A root node of the structure corresponds to a unique target that generates the feature vectors at its corresponding leaf nodes. For example, in Fig. 4 (b) there are 3 root nodes, and that means the algorithm can detect 3 targets from the 9 redundant images. The most-left tree in Fig. 4 (b) contains 1 root node, 4 leaf nodes and 2 middle-layer nodes which means that the images {D_1, D_2, D_3, D_4} are from the same real target represented by the root node. Additionally the structure indicates that {D_1, D_2} and {D_3, D_4} are taken from the platform#1 and #2 respectively.

The reason to pick the initial structure as in Fig. 4 (a) becomes clear in that we prepare for the worst case scenario where each leaf node is distinct from other leaf nodes. In other words, there are no duplicated images.

4. EXPERIMENTAL RESULTS

To demonstrate the DFG algorithm for data clustering and fusion, we use the sensor fusion as an example. Here we assume all parameters (means, covariance matrices, \( \omega_f \)) of DFG are known from the learning algorithm. We have 9 sonar images obtained from 3 real targets taken by 2 sensor platforms. Obviously, the 9 images are redundant since in the reality there are only 3 targets. We extracted 2 continuous-valued features (x-y location and x-y correlation length) from each image, each feature is 2D. The 2 features are plotted in Fig. 3. We also assume that the underlying generative model of the features is linear Gaussian. That is \( P(A = a | B = b) = N(\mu_a + \Sigma_a \cdot (b - \mu_b), \Sigma_a) \) if B is the only parent node of A, and \( P(B = b) = N(b; \mu_b, \Sigma_b) \) if B is a root node. From the experiment and by the experts, we know that a proper penalty function is \( pen(Z) = -2r - r^2 \) where \( r \) is the number of root nodes. The SA parameters are set as follows. \( K = 1, T_0 = 25, T_{lim} = 0.1, \alpha = 0.9 \).

The resulting optimal structure we got from the DFG in this experiment is illustrated in Fig. 4 which perfectly matches the ground truth in Fig. 3 (a) and (b). The posterior distribution at each root node describes the likelihood of the fused location. The \textit{maximum a posterior} (MAP) at each root node is actually the estimated (or fused) location of each group (plotted as the black star in Fig. 3). The covariance matrix of the posterior probability determines estimation uncertainty (plotted as the grey ellipse in Fig. 3). The number of iterations depends on the SA setting in which case the temperature schedule is shown in Fig. 5 (a). The run-time depends on the size of the network, and the number of parameters defining the underlying distribution. Here the run-time per iteration is about 0.8 sec.

5. CONCLUSION

DFG is a powerful data segmentation and fusion algorithm that is capable of modeling selection at the same time as data segmentation and data fusion. Furthermore, the entire framework is based on Bayesian estimation which is an ever-increasing desired property. The linear-time inference can be achieved on tree-structure network via the existing time-efficient sum-product algorithm. The framework can be implemented on the combination of DAG and UG since it is developed on the shoulder of FG. This satisfies the need for more complex models in real world application.

ACKNOWLEDGEMENT

The authors would like to thank J. Tory Cobb for the helpful discussion during the project and Office of Naval Research for financial support.

6. REFERENCES