Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

Saied Haidarian Shahri1, Majid Nili Ahmadabadi1, 2

1 Control and Intelligent Processing Center of Excellence, ECE Dept., University of Tehran
2 School of Cognitive Sciences, Institute for studies in theoretical Physics and Mathematics, Niavaran, Tehran, Iran
s.haidarian@ece.ut.ac.ir, mnili@ut.ac.ir

Abstract. Representing and modeling knowledge in the face of uncertainty has always been a challenge in artificial intelligence. Graphical models are an apt way of representing uncertainty, and hidden variables in this framework are a way of abstraction of the knowledge. It seems that hidden variables can represent concepts, which reveal the relation among the observed phenomena and capture their cause and effect relationship through structure learning. Our concern is mostly on concept learning of situated agents, which learn while living, and attend to important states to maximize their expected reward. Therefore, we present an algorithm for sequential learning of Bayesian networks with hidden variables. The proposed algorithm employs the recent advancements in learning hidden variable networks for the batch case, and utilizes a mixture of approaches that allows for sequential learning of parameters and structure of the network. The incremental nature of this algorithm facilitates gradual learning of an agent, through its lifetime, as data is gathered progressively. Furthermore inference is made possible, when facing a large corpus of data that cannot be handled as a whole.

1 Introduction

The human’s superiority, clearly, comes from its ability to learn. Although learning has many forms, some of which are shared among other creatures, only humans are able to build up complex hierarchies of ontology in the concrete and formal operational stages of psychological development. To reach this level of sophistication in artificial intelligence, the necessity of conceptualizing the knowledge and attending to important concepts cannot be overstated. Therefore we provide the relation of our proposed framework to attention abstraction, concept learning and cognition.

1.1 Attention Abstraction

Hidden variable networks can basically expose the concealed relations among a set of stochastic variables by observing a sufficient sample of the underlying observable process. Our proposed algorithm, which employs the information bottleneck
approach, can be utilized to extract the hidden phenomena from the original stochastic process. There are two intuitive interpretations for our attention behavior in this framework. Attention is like an information bottleneck apparatus which acts as a sifting device, purposively choosing among important regions of interest. The second interpretation is that attention is a hidden relation between the agent’s observations and actions which causes statistical dependency in what is observed by the agent. That is, attention is a hidden common cause between observations and actions, which sometimes is modeled as one variable augmented on both action and observation that accounts for their dependency.

What we attend to, most prominently, depends on the context of what we are trying to do. Therefore the actions we take and observations we perceive are not only related by what we attend to, but also related by the concepts that underlie the specific task we are carrying out. In short, attention can be seen as a component of the concepts that should be acquired to perform a task successfully. There are several reasons to think that a hidden variable network can learn useful concepts. The first reason is that concepts, like clusters, are an organization of observations with similar characteristics. Hidden variables are also a soft partitioning cluster of the observation space. The second reason is that when assuming the causal Markov assumption, there are two main reasons that a Bayesian network cannot learn the distribution faithfully (which means it cannot completely learn both the dependencies and independencies between the variables, described formally in Section 2). The two reasons are hidden common cause and selection bias (Figure 1). It is said that most of the real world domains can accuse the hidden common cause for not being able to learn a Bayesian network faithfully. By augmenting Bayesian networks with hidden variables we can assume the embedded faithfulness assumption.

![Graphical model](image1)

**Fig. 1.** Graphical models depicting (a) hidden common cause, and (b) selection bias.

The third reason is that augmenting with hidden variables is somewhat reminiscent of the role of mirror neurons [19], in that observing the state or action would provide the same clue to the underlying concept. This role is captured through the following dependency structure as shown in Figure 2. This means that knowing the action or the state will both contribute to knowing the cluster.

![Graphical model](image2)

**Fig. 2.** The relation between observation and action captured through a hidden variable.
Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

In a previous work we studied a concept learning architecture which consisted of a reinforcement learning agent along with an unrestricted Bayesian network. Concepts were considered a temporally extended sequence of state-action pairs which were learned from the sequence of actions taken and observations seen by the agent after it had learned a specific task with reinforcement learning. The task was for a mobile robot to learn wall following in a grid-world environment (Figure 3-right). The state consisted of its left and front sensors along with a history of its last two observations. By augmenting this state with the current action and a history of two previous actions a sample was formed. One beneficial feature of this algorithm was that it could seamlessly recognize the irrelevant features and attend only to features that contribute to predicting the next action by considering only the Markov blanket of the current action (Figure 3-left). However, this approach had some major drawbacks. The first problem was that it had to learn after a sufficient set of samples were provided, that is, it could only learn in the batch case. The second problem was that it could only predict Markov processes that did not possess any hidden phenomena. Therefore it was decided to create an algorithm that could resolve both issues.

Fig. 3. (left) The resulting Bayesian network for the wall following task. The dotted variables belong to the first history and the gray variables belong to the second history. (right) The robot in a grid world environment.

1.2 Concept Learning and Cognition

Our first inkling to this matter is probably the profound work of Piaget’s developmental theory. His pioneering work, later led to a close examination of various stages of child development, from infancy to adulthood, by many others. Here’s an example of a nine months old infant to elaborate on this issue.

“The infant plays with a toy that is then taken away and hidden under a pillow at the left. He raises the pillow and reclaims the object. Once again, the toy is taken and hidden, this time under a blanket at the right. The infant promptly raises, not the blanket, but the pillow again. So the relationships among objects are yet understood only in terms of pairwise transitions, as in the cycle of hiding and uncovering a toy. The intervention of a third object is not properly taken into account.” [3]

What suddenly springs to mind is the process of causality assimilation, and henceforth, the difference made to one’s mind or concepts, through this process.
Nevertheless, like all concepts, the notion of causality recapitulates some human experience. Does the concept of causation find its genesis in the observation of statistical relationships? "Adult thought might seem to provide a pre-established model, but the child does not understand adult thought until he has reconstructed it." [15]

To create a situated agent that can learn complex concepts like humans, and relate the effect of its actions to its observations and others’ actions, while acting and making decisions at the same time, the underlying process should be able to continually adapt to the environment and conceptualize the knowledge. In this paper, we advocate the use of graphical models as a means of causality assimilation, and hidden variables therein, to provide for abstraction. In the past few decades, there has been an exponential growth and interest in probabilistic graphical models. However, to date, very little effort has been focused on capturing its power for adaptive agents, which learn while living. By assuming hidden variables as concepts, that reveal the hidden common causes among observations, we describe an algorithm which learns the structure and parameters of a Bayesian network incrementally. This algorithm combines two methods of incremental learning with the batch approach to learning hidden variable networks, which benefits from the information bottleneck framework.

In the next section, we provide some definitions and the required notation for the rest of the paper. Sections 3 and 4 embark on the recent advancements in learning hidden variable networks for the batch case. Section 5 provides a view of the EM algorithm that justifies incremental, sparse and other variants of parameter learning for incomplete datasets. Section 6 describes incremental structure learning, and how it coincides with the EM algorithm, and wraps up with the proposed sequential learning of hidden variable networks algorithm. The last section discusses further extensions.

2 Background: Conditional Independence

Consider a set \( \mathcal{X} = \{X_1, \ldots, X_n\} \) of random variables, where each variable \( X_i \) may take on values from a finite set denoted by \( \text{Val}(X_i) \). We use capital letters such as \( \mathcal{X}, \mathcal{Y}, \mathcal{Z} \), for variable names and lower case letters \( x, y, z \), to denote specific values taken by those variables. Sets of variables are denoted by bold face capital letters \( \mathbf{X}, \mathbf{Y}, \mathbf{Z} \), and their instantiations by their bold lowercase counterparts \( x, y, z \). Now let \( P \) be a joint probability distribution over the variables in \( \mathcal{X} \) and let \( \mathbf{X}, \mathbf{Y}, \mathbf{Z} \) be subsets of \( \mathcal{X} \). The sets \( \mathbf{X} \) and \( \mathbf{Y} \) are conditionally independent given \( \mathbf{Z} \) if for all \( x \in \text{Val}(\mathbf{X}), y \in \text{Val}(\mathbf{Y}), z \in \text{Val}(\mathbf{Z}) \), either of the following conditions hold:

i. \( P(x \mid z, y) = P(x \mid z) \) whenever \( P(x, y) > 0 \).

ii. \( P(x \mid y) = 0 \) or \( P(y \mid z) = 0 \).

A Bayesian network is an annotated directed acyclic graph (DAG) that encodes a joint probability distribution of a set of random variables \( \mathcal{X} \). Formally a Bayesian network over \( \mathcal{X} \) is a pair \( B = \{G, \Theta\} \). The first component \( G \) is a DAG whose vertices correspond to the random variables \( X_1, \ldots, X_n \), and whose edges represent
Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

direct dependencies between the variables. The second component $\Theta$ represents the set of parameters that quantify the distribution. This component contains a parameter $\theta_{i\mid \text{pa}(x_i)} = P_b(x_i \mid \text{pa}(x_i))$ for each possible value $x_i$ of $X_i$, and $\text{pa}(x_i)$ of $\text{pa}(X_i)$, where $\text{pa}(X_i)$ denotes the set of parents of $X_i$ in $G$. A Bayesian network $B$ defines a unique joint probability distribution over $X$ given by:

$$P_b(X_1, ..., X_n) = \prod_{i=1}^{n} P_b(X_i \mid \text{pa}(X_i)).$$

2.1 Minimality, Faithfulness and Embedded Faithfulness Assumptions

If we create a causal DAG $G$ and assume that the observed probability distribution $P$ of the variables in the DAG satisfies the Markov condition with $G$, we say we are making the causal Markov assumption. A probability distribution $P$ satisfies the Markov condition with a DAG $G$ if the probability of each variable in the DAG is independent of its non-descendants conditional on its parents. Two of the most common problems of the Markov assumption are the hidden common cause and selection bias.

**Definition 1.** The DAG $G$ in the pair $(G, P)$ satisfies the **minimality condition** if it satisfies the Markov condition, and by removing any edge from $G$, the resultant DAG no longer would satisfy the Markov condition. Therefore it is a more demanding assumption than the Markov condition in the sense that any DAG satisfying the minimality assumption also satisfies the Markov condition, but not vice versa.

**Definition 2.** The DAG $G$ in the pair $(G, P)$ satisfies the **Faithfulness assumption** if it satisfies the Markov condition, and all conditional independencies in the distribution $P$ are entailed by the Markov condition in $G$.

The Faithfulness condition is quite strict and creating a DAG which satisfies this condition is not always possible. One way to deal with this problem is to enrich the process with hidden variables which remedies the most common problem in creating faithful DAGs, namely the hidden common cause, when the following assumption holds. Other than enabling us to create such a DAG, augmenting with hidden variables has more advantages and intuitive interpretations which will be evident through the rest of the paper.

**Definition 3.** A probability distribution $P$ over variables $X$ is **embedded faithfully** in DAG $G$ over $X = \{X, T\}$ if all and only the conditional independencies in $P$ are entailed by the Markov condition applied to $X$ and restricted to the variables in $X$, where $T$ is a set of hidden variables.

By creating a minimal DAG and assuming the embedded faithfulness assumption, we propose a sequential algorithm to learning hidden variable networks.
2.2 Structure Learning with Complete Data

There are two basic approaches to learning the structure of Bayesian networks. The first approach is a constraint based approach that uses independence tests directly [16]. Based on some statistical test a set of independence clauses is formed and a network that captures this set to the maximum extent possible. The high sensitivity of this method to statistical tests requires that all the data be available before hand. Therefore this method is not suitable for incremental learning.

The second method in learning the structure of Bayesian networks is the common score based approach. There are two important aspects in this approach, the scoring function and the search procedure. The score is defined to measure the compatibility of the model to the data.

Since the number of possible structures of a DAG is super-exponential, the search procedure is usually greedy with some local operators (e.g. add edge, reverse edge, delete edge) that change the structure. This search is usually guided by a scoring function such as the minimum description length (MDL) principle [12] based score, or the Bayesian score BIC/BDe [10]. These scoring functions have an important property that when the data \( D \) is complete, they are decomposable to the sum of the local contributions of each variable \( X_i \).

\[
\text{Score}(g, D) = \sum_i \text{FamScore}(X_i, \text{Pa}_i | D) \tag{1}
\]

The \text{FamScore} term depends only on values of \( X_i \) and \( \text{Pa}_i \) in the training instances. In particular, the BDe score is defined as

\[
\text{Score}_{\text{BDe}}(g, D) = \sum_i \sum_{\text{pa}_i} \left( \log \frac{\Gamma(\alpha(\text{pa}_i))}{\Gamma(N(\text{pa}_i)) + \alpha(\text{pa}_i))} + \sum_{x} \log \frac{\Gamma(N(x_i, \text{pa}_i) + \alpha(x_i, \text{pa}_i))}{\Gamma(\alpha(x_i, \text{pa}_i))} \right)
\]

where \( \Gamma \) is the Gamma function and the \( \alpha() \) terms are dirichlet hyper-parameters of the prior distributions over the parameterization terms, and \( N() \) are the corresponding empirical sufficient statistics.

3 Multivariate Information Bottleneck

The information bottleneck method [17] is a generalization of rate distortion theory which formally tackles the unsupervised non-parametric data clustering problem. This approach yields a set of self consistent equations whose solutions can be found by a convergent re-estimation method.

3.1 The Information Bottleneck

Suppose \( Q(Y, X) \) is a joint distribution of two random variables. The information bottleneck tries to extract the relevant information that \( Y \) contains about \( X \) by
Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

partitioning the space of \( Y \) into coarser distinctions that are still informative about \( X \). For example, assume that we want to partition the words \( (Y) \) appearing in several documents in a way that is most relevant to the topics \( (X) \) of these documents.

To achieve this goal, it uses the mutual information measure between two random variables,

\[
I_q(X; Y) = \sum_{x,y} Q(x, y) \log \left( \frac{Q(x, y)}{Q(x)Q(y)} \right)
\]

Introducing a new variable \( T \), it provides the bottleneck relation between the two variables which trades off the compression that the problem requires by the preservation of the relevant information. Formally it attempts to balance this tradeoff by minimizing the Lagrangian

\[
\mathcal{L} = \sum_{i} \beta I_q(T; X)
\]

where \( \beta \) controls the tradeoff. Tishby et al. [17] show that the optimal partition for a given value of \( \beta \) satisfies the set of equations

\[
Q(t | y) = \frac{Q(t)}{Z(y, \beta)} \exp \left\{ -\beta D(Q(X | y) || Q(X | t)) \right\}
\]

for all \( t \) and \( y \), where

\[
D(P(X) || Q(X)) = \sum_x P(x) \log \left( \frac{P(x)}{Q(x)} \right)
\]

is the Kulback Leibler divergence between the distributions \( P \) and \( Q \) over the set of random variables \( X \).

3.2 The Multivariate Case

The multivariate extension of this framework [7] makes it possible to model the interaction of multiple observed variables using several bottleneck variables. These interactions are represented with two Bayesian networks. The first, called \( G_{in} \), represents the required compression, and the second, called \( G_{out} \), represents the independencies that we are attempting to model between the bottleneck variables and the target variables.

A hypothetical and naïve example of the two networks is depicted in Figure 4, which \( G_{in} \) specifies that \( T \) is a stochastic function of its parent in the graph \( Y \), and \( G_{out} \) specifies that we want \( T \) to make \( Y \) and the variables \( X_i \)'s independent of each other. Formally, the extension of Friedman attempts to minimize the Lagrangian

\[
\mathcal{L} = I_{\text{out}}(G_{in}, G_{out}) = I_{\text{out}} - \beta I_{\text{out}}
\]

where

\[
I_{\text{out}} = \sum_i I(X_i; \mathbf{Pa}_{G_{out}}^G)
\]

Friedman et al. [2001] also present an analogous variational principle that reformulates the problem as the tradeoff between compression of mutual information in \( G_{in} \) so that the bottleneck variables \( T \) would help us model a joint distribution that is as close as possible to the form of a target Bayesian network \( G_{out} \). Formally it attempts to minimize the following objective function
\[ L^2(Q, P) = I_Q(Y; T) + \gamma D(Q(Y, X, T) \| P(Y, X, T)) \] (2)

where \( Q \) and \( P \) are joint probability distributions that can be represented by the networks of \( G_{in} \) and \( G_{out} \) respectively. The two principles are analogous under the transformation \( \beta = \gamma/(1 + \gamma) \), assuming \( I^\gamma_{in} = I_Q(Y; T) \). This analogous formulation is later used to develop a learning algorithm for hidden variable networks.

**Fig. 4.** Definition of \( G_{in} \) and \( G_{out} \) for Multivariate Information Bottleneck framework. \( G_{in} \) encodes the distribution \( Q \) that compresses \( Y \). \( G_{out} \) encodes the distribution \( P \) that we want to approximate using \( Q \) [4].

### 4 Information Bottleneck Expectation Maximization

Due to lack of space we will briefly review the important results which lead to a learning algorithm of hidden variable networks [4] for the batch case and elaborate on using this method to learn incrementally.

Consider a set of random variables \( X \) which is augmented with a deterministic instance identity variable \( Y \), and the set of hidden variables \( T \). The instance identity variable is to enumerate the training samples. Elidan and Friedman [2005] show that if \( G_{in} \) is a Bayesian network such that all of the variables \( T \) are independent of \( X \) given \( Y \), and \( G_{out} \) is a Bayesian network such that \( Y \) is a child of every \( T \) and has no other parents, then except for using a factorization that is reminiscent of the mean field variational approximation [11] in favor of computational efficiency, the multivariate information bottleneck Lagrangian (2) can be formulated as

\[
L^2_{EM} = \sum_i I_Q(T_i; Y) + \gamma \left( E_Q[\log P(X, T)] - \sum_i E_Q[\log Q(T_i)] \right)
\]

where the + superscript denotes the approximation used. As stated earlier there exists a set of self consistent equations that iteratively converge to a local maximum of the Lagrangian. Only the difference for the multivariate case here, is that when using the mean field factorization, there is an independent set of equations for each variable \( T_i \),

\[
Q(t_i | y) = \frac{1}{Z(t_i, y, \gamma)} Q(t_i)^{1-\gamma} \exp \{\gamma EP(t_i, y)\}
\]

where

\[
EP(t_i, y) = E_Q[\log \frac{P(x[y], T)}{P(x[y], T_i)}]
\]

and \( Z(t_i, y, \gamma) \) is a normalizing constant that equals to
Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

\[ Z(i, y, \gamma) = \sum Q(t_i) \exp \{ \gamma EP(t_i, y) \} \]

One should note that the expectation in $EP(t_i, y)$ is over the markov blanket of $T_i$, and also all the other terms that do not depend on $T_i$ can be averaged out in the normalization factor and not be calculated. Therefore $EP(t_i, y)$ can be computed efficiently.

Interestingly, there is a close relation between the information bottleneck and expectation maximization methods. Elidan and Friedman [2005] also show that the information bottleneck method is, in a way, a deterministic annealing approach to expectation maximization, which regularizes the expected empirical distribution surface and helps to create improved generative models.

To explore this connection, an alternative EM formulation introduced by Neal and Hinton [13] is employed which uses a quasi-static approach to combine the expectation and maximization steps in one functional. It replaces the expectation step of $Q(X)$ by maximizing an auxiliary distribution which is the completed empirical distribution $Q(X, T)$. Using the above notation, the functional can be formulated as

\[ \mathcal{F}_{\text{EM}}[Q, P] = EP[\log P(X, T)] - H_P(T | Y) \]

where $H_P(T | Y) = EP[-\log Q(T | Y)]$, and $Q(X, Y)$ is fixed to be the observed empirical distribution. Furthermore the EM iterations correspond to maximizing $\mathcal{F}_{\text{EM}}[Q, P]$ with respect to $Q(T | Y)$ while holding $P$ fixed, and then maximizing $\mathcal{F}_{\text{EM}}[Q, P]$ with respect to $P$ while holding $Q(T | Y)$ fixed. Elidan and Friedman [2005] proved the following relation between this functional and the so called IB-EM Lagrangian

\[ \mathcal{L}_{\text{EM}} = (1 - \gamma)I_Q(T, Y) - \gamma \mathcal{F}_{\text{EM}}^* [P, Q] \]

where $\mathcal{F}_{\text{EM}}^* [Q, P]$ is the decomposition of the EM functional according to the mean field factorization. As a result, minimizing the IB-EM Lagrangian is equivalent to maximizing the EM functional combined with an information theoretic regularization term, which is independent of the generative distribution $P$ that we are striving to model. When $\gamma = 1$, the solution of the Lagrangian and the EM functional coincide and finding a local minimum of $\mathcal{L}_{\text{EM}}$ is equivalent to finding a local maximum of the likelihood function.

5 Incremental Expectation Maximization

When there are missing variables in your data, the parameters of the model become dependent, and there are no closed form solutions for the marginal likelihood. The conventional methods to deal with this problem are to use Monte Carlo approaches [14] and large sample approximations [1]. Although Monte Carlo methods are accurate, they are very computationally inefficient. Estimating the marginal likelihood with large sample approximations requires computing the MAP/ML state of the parameters that can be found either by gradient based optimization or Expectation-
Maximization method [2]. The Expectation-Maximization method is better suited to our problem, since it readily lends itself to an incremental setup.

The EM algorithm finds the maximum likelihood estimates of the parameters of a model when there are missing or unobserved variables. By starting at an initial guess, it iteratively 1) finds the distribution of the unobserved variables (E-step) given the known values for the observed variables and the current parameter estimates, and 2) maximizes the likelihood of parameter estimates (M-step) under the assumption that the distribution found in the E-step is correct.

There is a more general class of algorithms which include the EM as a special case, in which the M-step of the algorithm can be only partially implemented, with the new estimate for the parameters improving the likelihood given the distribution found in the E-step. As Dempster et al. has shown, such a partial M-step in the Generalized EM (GEM) always results in the true likelihood improving as well.

In order to prove the convergence of this algorithm we will use a view of the EM algorithm introduced by Neal and Hinton [13] that justifies incremental, sparse and other variants. It has been previously shown that the M-step can be implemented partially. Now the only problem to resolve is the E-step. In this viewpoint, both the E and the M steps are seen as maximizing the joint function of the parameters and of the distribution over the unobserved variables, which also lead to a maximum of the true likelihood. In light of this, maximizing the joint EM functional \( F_{EM}[Q,P] \) defined earlier is equivalent to the original iterative EM algorithm as the following theorem shows.

**Theorem 1** (Neal and Hinton, 1998) If \( F_{EM}[Q,P] \) has a local maximum at \( (Q',P') \), then the log likelihood function \( E_Q[\log P(X)] \) has a local maximum at \( P' \) as well. Similarly, if \( F_{EM} \) has a global maximum at \( (Q',P') \), then \( E_Q[\log P(X)] \) has a global maximum at \( P' \).

**Corollary 1** Any quasi-static approach, including the incremental methods that partially improve the expected distribution of parameters over the unobserved variables, which contribute to the optimization of the EM functional \( F_{EM}[Q,P] \) will converge to a stationary point of the observed log likelihood function \( E_Q[\log P(X)] \).

Corollary (1) shows that the E-step of the algorithm can be partially implemented as well. In particular, the incremental setting of this algorithm converges faster to the true likelihood in terms of the number of E and M steps, since any minor improvement in the E-step immediately affects the M-step, and it does not have to wait for a complete maximization of the E-step. Although, note that this fact does not mean a gain in computational speed, because computational speed depends on the complexity of the E and M steps individually [13].

### 6 Incremental Structure Learning

Learning the structure of a Bayesian network as discussed in section 2.2 is much easier when training instances are complete than when there are hidden variables or missing data. It is possible to use the EM method in the previously described setting with the methods of section 2.2. First, one must make all the neighboring DAGs of
Learning to Attend to Concepts: An Incremental Hidden Variable Networks Approach

the current best model, and then complete the data according to that model with the EM method, to make scoring each model possible.

Although this is possible, it is very time consuming and computationally inefficient. Another more tractable solution to this problem is described in [8]. The Structural Expectation Maximization (SEM) algorithm interleaves the structural changes in the search procedure with the M-step of the EM algorithm in order to mitigate the burden of having to score several incomplete models. The expected speedup of this approach is significant. This method reduces the complexity of the algorithm by an $O(n^2)$ factor, which results in a computationally efficient approach to this problem. [9] also extends the Structural EM algorithm to adapt it for Bayesian model selection and proves its convergence.

There is still another problem in learning the structure with the current approach, which basically is, how the sufficient statistics are managed. When learning the structure incrementally this can be a very delicate task. In the rest of this section we will employ Friedman and Goldszmidt’s [6] method to resolve this problem. As previously stated, learning the structure consists of a search procedure and a scoring function. The two major commonly used scoring functions are the Bayesian score, and the one based on the minimum description length (MDL) principle. The details of these scores for the batch learning case can be found in [10] and [5] respectively. It is proven that these scores are asymptotically equivalent up to a constant number as the sample size increases. In addition, they are asymptotically consistent, that is, with probability equal to one (almost surely) the learned distribution converges to the underlying distribution as the number of samples increases.

When data is complete, both scores have two appealing properties. The first property is that for a fixed network structure $G$, there is a closed form formula for finding the optimal parameters that maximize the score. Moreover, these optimal parameters can be extracted from the sufficient statistics for the structure $G$. To understand how sufficient statistics relates to the problem, consider the function

$$N^\varnothing_X(x) = \sum_{x \in \mathcal{D}} 1\{X = x\}$$

where $\mathcal{D}$ is the set of complete data, and $1\{\cdot\}$ is similar to the kronecker’s delta function which is 1 whenever it is true and zero otherwise. $N^\varnothing_X(x)$ is called the sufficient statistics of data $\varnothing$ for $X = x$. Let $\hat{N}^\varnothing_X$ be the vector of sufficient statistics for all $X = x$. As previously mentioned, the optimal choice of the parameters $\theta_{X,pa}$ is a function of $\hat{N}_{X,pa}$. The previous batch approach to maximizing the parameters is to keep track of all the unique instantiations of the variables $X$ and store every count. But clearly, the number of sufficient statistics is exponential in the number of variables. Thus, storing the count for every unique instantiation is not possible, especially when there is a massive amount of data.

The second property is decomposability (1) of the score given a fixed structure. Let a family be composed of the variable $X_i$ and its parents. Then, for a given structure $G$, one should only store the sufficient statistics $\hat{N}_{X_i,pa}$ for the set of families of $G$. Therefore, when searching for the best structure using either of the scores, only the sufficient statistics for the set of families of structures in the search space is needed.
Let $Suff(G)$ denote the set of sufficient statistics for graph $G$, that is, $Suff(G) = \{ \hat{N}_{x_i, pa_i} \mid 1 \leq i \leq n \}$. Unlike parameter optimization in which an update can be done after every observation, structural changes need a reasonable amount of data, to be scored confidently. Consider a setting where after every $k$ observations there is a structural update. A naive way to deal with the exponential number of sufficient statistics is to only store the counts for the best model. Unfortunately, by using the so called MAP model as the prior for the next learning iteration, much information is lost and the learning process is strongly biased towards the MAP model.

Another elegant approach to obviate the bias is to consider several models instead of one best model. In particular, by defining a search frontier for the search procedure used, this paradigm can be generalized. Let $F$ be the networks in the next search frontier. Now, if we keep track of all the sufficient statistics $S$ such that $S = Suff(G) \cup \bigcup_{G'} Suff(G')$

where $G$ is the initial structure in the search procedure, we can tradeoff between the amount of information kept and the quality of the learned network. The interesting fact about this approach is that, if all the networks that can be scored by $S$ are denoted by $Nets(S)$, that is, $Nets(S) = \{ G \mid Suff(G) \subseteq S \}$

then $F \subseteq Nets(S)$. In general $Nets(S)$ contains many networks not in $F$.

There is another issue that needs to be discussed. In the model selection problem for the batch case, we rely on the score to evaluate different models according to the same dataset. But in the current setting, the underlying problem is a general model selection problem, where we compare models $M_1$ and $M_2$, such that model $M_1$ is evaluated with respect to training set $D_1$, while model $M_2$ is evaluated with respect to training set $D_2$. This happens because of the way that the sufficient statistics are managed. Therefore the current scoring functions are not suitable. [6] show how the scoring functions should be changed and justify their claim for the MDL score. Evidently, it would be true for the BDe score as well, since they are asymptotically equivalent. According to our notation, they suggest that the BDe score be as the following

$$Score_{\text{ms}}(X_i, Pa_i) = \frac{\sum_{N(x_i, pa_i)} \text{FamScore}(X_i, Pa_i)}{\sum_{x_i, pa_i} N(x_i, pa_i)},$$

where $\text{FamScore}$ is the local contribution of the family to the score.

To sum it up, by combining the methods described above, the main loop of the sequential hidden variable network algorithm would be as follows:

**Set** $G$ to be an initial network  
**Let** $F$ be the initial search frontier for $G$  
**Let** $S = Suff(G) \cup \bigcup_{G'} Suff(G')$  
**For all** $\hat{N}_x \in S$  
$N(x) \leftarrow N_0 \cdot P_G(x)$  
**Forever**  
Read data instance $y$
Learning to Attend to Concepts: An Incremental Hidden Variable Networks

For all $\hat{N}_x \in S$

$N(x) \leftarrow \alpha \cdot N(x) + P(x \mid y)$

If $n \mod k = 0$

Let $G = \arg \max_{G \in \mathcal{G}(S)} \text{Score}(G' \mid S)$

E-step: Maximize $-\mathcal{L}_{EM}$ by varying $Q(T \mid Y)$

Compute the expected sufficient statistics.

M-step: while holding $Q$ fixed,

- Search for the structure $G_{out}$ of $P$ that maximizes $\text{Score}_{\text{mle}}(G \mid S)$ using the expected sufficient statistics of $Q$.

- Maximize $-\mathcal{L}_{EM}$ by varying the parameter of $P$ using the structure $G_{out}$ selected.

Update the frontier $F$

Update $S$ to $\text{ Suff}(G) \cup \bigcup_{G' \in F} \text{ Suff}(G')$

Compute optimal parameters for $G$ from $S$

Output $G$

7 Discussion

In this paper we developed an incremental learning framework so that the agent could attend to important concepts while learning them gradually. Although all the variables are assumed to be discrete in the current algorithm, it can be extended to the continuous case and exponential family distribution. It can be seen from the recent work in attention literature [18] that basically, context is very important for attention, providing information in the sense of how to perform top-down priming on feature based interpretation, and hidden variables learned via Bayesian interpretation, in particular, provide these contextual clues that will consequently serve attention in terms of information selection by providing top-down information. Furthermore, selection of concepts and the relevant attention shifts that are important at a certain time instant are driven by the learned structure proposed in this paper.

Acknowledgments

The authors would like to thank Daphne Koller and Gal Elidan for their early insight into this work. Also Hamid Haidarian’s inexplicable effort for his critical review is genuinely appreciated.

References


