Exploiting Task Relatedness to Learn Multiple Bayesian Network Structures

Diane Oyen
Computer Science Dept.
University of New Mexico
Albuquerque, NM 87131

Terran Lane
Computer Science Dept.
University of New Mexico
Albuquerque, NM 87131

Abstract

We address the problem of learning multiple Bayesian network structures for experimental data where the experimental conditions define relationships among datasets. A metric of the relatedness of datasets, or tasks, can be described which contains valuable information that we exploit to learn more robust structures for each task. We represent the task-relatedness with an undirected graph. Our method uses a regularization framework over this task-relatedness graph to learn Bayesian network structures for each task that are smoothed toward the structures of related tasks. Experiments on synthetic data and real fMRI experiment data show that this method learns structures which are close to ground truth, when available, and which generalize to holdout data better than an existing multitask learning method, learning networks independently, and learning one global network for all tasks.

1 INTRODUCTION

Bayesian networks give a compact representation of relationships among variables in data. Thus they are often used to model underlying structure in experimental data. This data may be collected under various experimental conditions to explore how the structure of the Bayesian network changes across conditions. Furthermore, conditions can be combined to further explore the effects on network structure in various contexts.

For example, the functional network of the human brain may be observed while performing different tasks, such as reading or listening. From these two conditions, four different task contexts could be realized: reading and listening at the same time, reading alone, listening alone, and the control environment. Thus, we could learn four different networks, but we do not expect these networks to be independent of each other. The reading & listening network likely includes structures found in both the reading network and the listening network. However, not all contexts are directly related.

Often the amount of data available to learn these networks is limited, thus it is advantageous to leverage the full dataset across the contexts to learn these similar networks. Furthermore, our approach has the ability to estimate networks under contexts for which there is little or no data at all available by using data from similar contexts. Exploitation of the similarity among contexts is especially useful in studies where it is difficult to directly manipulate conditions, such as in estimating functional brain networks for schizophrenic patients under various medications, as there could be many combinations of conditions with little data.

2 GENERATIVE MODEL

2.1 CONTEXT SPECIFIC NETWORKS

Bayesian networks model the probabilistic relationships among random variables in a system. Conditional independence between two variables is represented by the lack of an edge between the nodes representing those variables. These relationships may change as the context under which the system operates changes, i.e. there is a network specific to each context.

We could learn a functional brain activation network for the activity 'reading' and a separate network for 'listening'. What if the person is reading and listening at the same time? We would expect that the functional brain network elicited by this experiment would have something in common with both the networks for listening and reading, but would not be identical to either of them. Furthermore, we do not expect that the listening and reading networks are directly related (although it is quite likely that they are indirectly related). To learn all three of these networks, we would like to leverage the listening data while learning the network for listening and reading; and indirectly leverage the listening data while learning the network for reading.

2.2 CONDITIONS AND CONTEXTS

A **condition** is a binary variable of the experimental paradigm, and a **context** is a setting of all available conditions. For example, brain images could be collected under three possible conditions, such as reading, listening, and speaking which we will represent with $Cond \in \{R, L, S\}$. These conditions are not mutually exclusive, thus there are eight possible contexts $Cont \in \{111, 110, 101, 011, 100, 010, 001, 000\}$ where each bit is an assignment to R, L, S representing the presence (1) or absence (0) of that condition.

Clearly, the number of contexts is exponential in the number of conditions. In this paper, we explore only problems with three or fewer conditions to avoid this exponential blow-up in complexity. However, extending our method to handle larger numbers of conditions is a direction for future work.

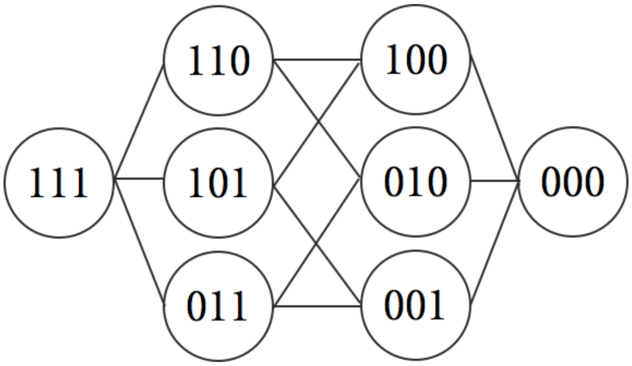


Figure 1: The metagraph: vertices are contexts, edges are similarity between contexts.

2.3 RELATING CONTEXTS

Intuitively, we can define two contexts to be similar if their condition settings differ by one bit (the Hamming distance). Using the current example with three conditions, this similarity metric induces the hypercube topology of Figure 1 when drawn as an undirected graph. The vertices are the contexts, and the edges represent the similarity between contexts. We call this the *metagraph* because it represents the similarity relationships among the graph structures of each context., Figure 2.

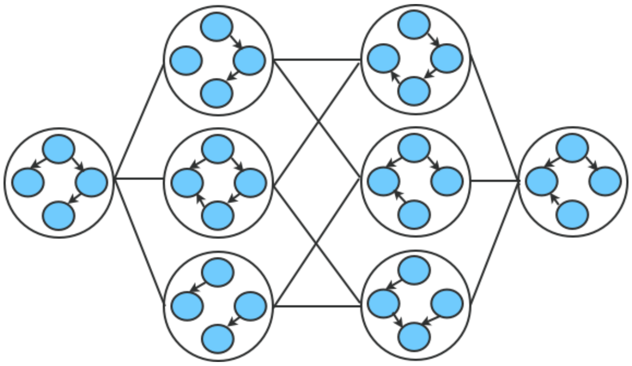


Figure 2: Learned structures vary smoothly across the metagraph.

2.4 SMOOTHLY VARYING STRUCTURES

In our brain network example, the metagraph has a lattice structure but this is not a requirement for our algorithm. Any distance metric could be used, thus the metagraph could be any undirected graph including weighted graphs. As two examples: a fully connected graph would be the case where all graphs are smoothed together, the typical multi-task learning problem; and, a completely disconnected graph would mean learning each structure completely independently from each other. The key challenge is to learn network structures for each context such that those structures vary smoothly across the metagraph. There is an inherent tradeoff between learning structures that closely fit the data for each context (as learning independently would do) and learning structures that are the same between neighboring contexts (as pooling the data would do).

3 BAYESIAN NETWORK STRUCTURE SEARCH

A Bayesian network $B = \{G, \theta\}$ describes the joint probability distribution over n random variables $X = \{X_1, X_2, ..., X_n\}$, where G is a directed acyclic graph (DAG) and the conditional probability distributions are parameterized by θ (Heckerman et al., 1995). An edge (x_i, x_j) in G represents a direct probabilistic dependency from the parent x_i to the child x_j. The *structure* of the Bayesian network, G, is of particular interest in many domains as it is easy to interpret and gives valuable information about the direct interaction of variables.

The structure G and parameters θ can be learned from a set of data $D = \{x_1, x_2, ..., x_m\}$ where each x_i is a complete assignment of values to variables $X_1, X_2, ..., X_n$ (Heckerman et al., 1995). For any D and a fixed G, θ can be directly estimated using the maximum likelihood estimator (MLE). The more difficult problem is learning G, which is typically an heuristic search for the G that maximizes the likelihood of the data D. Preference is generally given to simpler structures (fewer edges) as overly complex structures can be overfit to the data as well as being more difficult to understand. Therefore, a score such as BIC or BDe is often maximized which favors simpler structures.

One reasonable heuristic is greedy search which starts from an initial structure and then iteratively makes the single best change (edge addition, deletion or reversal) to the network structure until no further improvements can be made. The best change is the one that gives the greatest improvement in score.

4 LEARNING MULTIPLE BAYESIAN NETWORKS

For multiple sets of data D_i, multiple networks G_i can be learned by optimizing each score independently.

$$\mathit{TotalScore}_{indep} = \sum_i \mathit{Score}(G_i|D_i) \tag{1}$$

To take advantage of the information given by the relatedness of tasks, we introduce a regularization term (L) that encourages smooth variations in structures across the metagraph. For example, a first-order regularizer would penalize differences in structure between neighbors in the metagraph, as in Equation 3. The parameter, α, weights the relative importance of smoothness across structures versus the fit to data of the individual structures.

$$\mathit{TotalScore}_{smooth} = (1-\alpha)\left(\sum_i \mathit{Score}(G_i|D_i)\right) - \alpha L \tag{2}$$

$$L = \sum_i \sum_{G_j \in Ne\{G_i\}} \mathit{dist}(G_i, G_j) \tag{3}$$

Any graph distance metric can be used for $\mathit{dist}()$ depending on the desired definition of structure similarity. In the case where the datasets come from analogous random variables, the distance metric can be a simple edit distance in the number of edge additions, deletions, and reversals necessary to change network G_i into network G_j. Note, however, that the learned networks need not contain the same set of vertices as long as an appropriate graph distance is used.

The beauty of this algorithm is in its flexibility to address a wide variety of problems. We can address fairly complex relationships among related tasks through the metagraph. The regularization function can take any form that operates in the domain of graphs. Finally, the distance metric between structures which is being smoothed can take any form that works in the space of DAGs.

5 RELATED WORK

Multitask learning addresses the problem of learning models for several related tasks, usually by inducing transfer of task-related information by iteratively learning models for each task biasing the models toward each other (Caruana, 1997). (Niculescu-Mizil and Caruana, 2007) solve the particular problem of multitask Bayesian network structure learning using the structure of each learned model as a prior on the other structures. Our primary difference is in the formulation of the problem. Rather than assuming all tasks are related, we allow a metric of task-relatedness to dictate which tasks transfer information. Rather than using Bayesian priors, we use regularization because it gives a convenient framework for formulating smoothness across related tasks. We show that exploiting more complex models of task-relatedness is important by comparing our method against (Niculescu-Mizil and Caruana, 2007).

(Nassar et al., 2008) apply multitask Bayesian network learning to multiorganism gene network estimation. Their method is a special case of (Niculescu-Mizil and Caruana, 2007) for learning only two related networks with an automated method for learning the strength parameter of the prior. (Luis et al., 2009) give a transfer method for learning both structure and parameters of related Bayesian networks. Transfer learning is somewhat different than the problem that we are addressing in that it assumes a good model is learned or known for one task and information from that model is transferred to a new task. In multitask problems, we learn models for all tasks simultaneously.

Our method requires that the relatedness of tasks be defined by a graph. This idea of modeling task relatedness through graphical models was explored by (Eaton et al., 2008). Their work could be used to create this task-relatedness graph if it is not already known. This paper only addresses the case where the task-relatedness graph is created by a domain expert rather than through automated learning.

As the number of conditions increases, the number of contexts, and hence the metagraph, grows exponentially. (Yackley et al., 2008) use a kernel on a metagraph to approximate the space of metrics represented by the graph. We did not address large context spaces in this paper, but their method could be incorporated in future work to address this problem.

6 EXPERIMENTS

6.1 SYNTHETIC DATA

Synthetic data allows us to control ground truth to test the properties of our algorithm.

Synthetic data was created for various numbers of conditions. Given c conditions, there are 2^c contexts, which are related by the previously defined metagraph (the Hamming cube). To create networks that are related through this metagraph, a cascading approach to generate each network is taken. The generative process first creates a random DAG and associates that DAG with one of the nodes of the metagraph. Now, for each neighbor of that node in the metagraph, another random DAG is selected with the following edge probabilities: $P(v_i, v_j) = \mathit{smoothness} * \sum_{G_{par} \in Parents(G)} G_{par}(v_i, v_j) + (1 - \mathit{smoothness}) G_{independentlyRandom}(v_i, v_j)$. The chances of edges being shared between neighboring structure varies with the value of the smoothness parameter. If the smoothness parameter is 1, then all graph structures will be identical to the first randomly generated DAG. If the smoothness parameter is 0, then each structure is independently randomly selected.

In these experiments, each DAG has 10 binary vertices. Random DAGs are generated by randomly selecting a node order and then randomly assigning edges, with a maximum fanin of two. However, when graphs are combined no limit is placed on fanin. Once the structure is determined, CPTs are generated randomly using a $\mathit{Dirichlet}(10, ..., 10)$ prior.

6.2 FMRI DATA

The brain can be thought of as a collection of regions of interest (ROI), where each of these regions performs a specific function and communicates with other ROIs to produce thought and behavior. Modeling brain activity data as a Bayesian network gives insight to how those ROIs interact. Functional magnetic resonance imaging (fMRI) produces a 3-dimensional activity map of a subject's brain over time. Individual voxels from the 3-dimensional brain volume can be aggregated into ROIs by mapping an atlas onto the volume. Thus activity levels are recorded for each ROI at each time step. fMRI studies often have an experimental paradigm where subjects respond to stimuli (usually auditory or visual). Given a paradigm where various stimuli occur concurrently, it is useful to learn how each stimulus affects the brain network individually and in concert with other stimuli.

For the 2006 Pittsburgh Brain Activation Interpretation Competition (PBAIC) subjects view two similar movies and report what they see and hear during the movies (Schneider and Siegle, 2006). Also provided are 16 ROIs which have been identified as relevant to watching movies (Schneider and Siegle, 2006, Appendix E). The goal of the competition is to predict what is happening in the movie given only fMRI data. However, we are more interested in learning the structure of the functional brain network under different conditions. Therefore we learn Bayesian networks to model the functional brain network. Results are given for the prediction problem to show that we are not losing information by imposing our task-relatedness metric on this data.

In the movies there are periods of time where spoken language is present and where tools are present, which can be considered two conditions (presence of language and presence of tools). Given these two conditions, there are four contexts representing the possible combinations of these conditions, which we label {00, 01, 10, 11} for the setting of Language/Tools. We segmented the fMRI data into periods for each context, giving four related datasets.

To test algorithms, a random selection of 50% of the training data from one subject (Subject 2) was chosen from the data of the first movie, then the learned structures were scored against the data from the second movie. 10 different random training sets were used to give significance values when comparing methods. These scores were compared by paired t-test to give significance results.

6.3 ALGORITHMS FOR COMPARISON

We tested our algorithm (**Smooth**) see how well it performed compared to other algorithms. First we tried two naive algorithms: learning each network independently (**Independent**) and learning a single network for all contexts (**Global**). These naive algorithms give a baseline comparison of the two extremes (assuming complete independence and assuming complete similarity). We also compare against a multitask Bayes net structure learning algorithm (**Multitask**) (Niculescu-Mizil and Caruana, 2007). This algorithm assumes a generative model in which all tasks are related rather than the Hamming distance model that our algorithm assumes.

7 RESULTS

7.1 METHODS

A head-to-head comparison is performed between our algorithm and several other methods. For each test with synthetic data, ten sets of synthetic graphs and corresponding datasets were generated. Each algorithm learned a set of structures from each of these ten datasets and the structures were scored on a large set of test data.

The score varies greatly between different generated graphs. This variance across training data is larger than the variance of scores across learning methods for a given set of training data. We cannot simply average scores across the 10 training sets and then compare between learning methods because the variance across datasets washes out the differences between learning methods.

Instead a paired t-test is performed between our learning method and each of the other learning methods. The paired t-test makes a head-to-head comparison between the learning methods across the 10 datasets to determine whether the differences are significant (p=0.05). Each point in the graph shows the difference in score between our learning algorithm and one other learning algorithm averaged across the datasets. Thus the differences are averaged, not the raw scores. A **positive value** indicates that our algorithm has better scores. A **negative value** indicates that the other algorithm has better scores. Insignificance is marked with a *.

7.2 LEARNING STRUCTURES THAT GENERALIZE TO TEST DATA

Our primary goal is to learn structures that accurately describe the distribution of data. To test the appropriateness of our learned structures, we compare the structure score against holdout data to demonstrate how well the learned structures generalize to new data from the same generating process.

7.2.1 Training Set Size

Our algorithm, **Smooth**, outperforms both **Independent** and **Global** on synthetic data when the amount of training data is limited (Figure 3(a,b,c)). As expected, **Smooth** is able to leverage information from across the related contexts to learn a better model than **Independent**, particularly as the amount of training data is very small. Yet, **Smooth** is also able to differentiate between contexts, learning structures better fit to each particular context than **Global**.

7.2.2 Single Context with Little Data

We test the case where there is little or no data in one particular context. Clearly with no data, the best that **Independent** can do is to select the most likely graph given the prior. Figure 3 (d,e,f) shows the results for a single context with varying amounts of training data while all other contexts have $N_{samples} = 1000$. **Smooth** outperforms **Independent** most of the time, especially when the available data is very small. It is never significantly worse to use **Smooth** compared to **Global** when there is little data for a single context. However, if there are only two contexts (one condition), then the differences between **Smooth** and **Global** are often insignificant.

7.2.3 Comparison to Multitask Learning

Smooth always outperforms **Multitask** (Figure 4) on synthetic data. This is not surprising as data was generated using the same generative process assumed by **Smooth**. Results are shown for various training set sizes with synthetic data drawn from one, two, and three conditions.

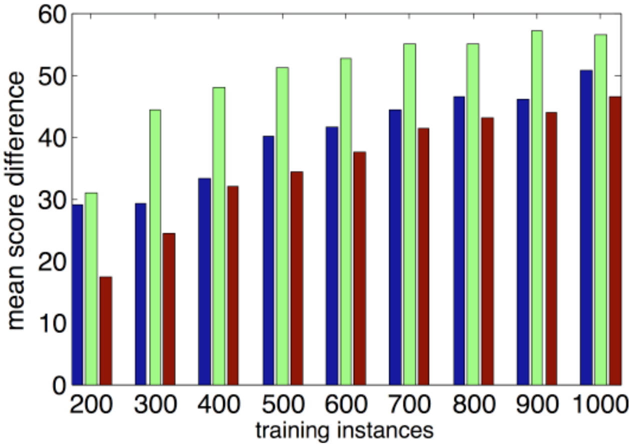


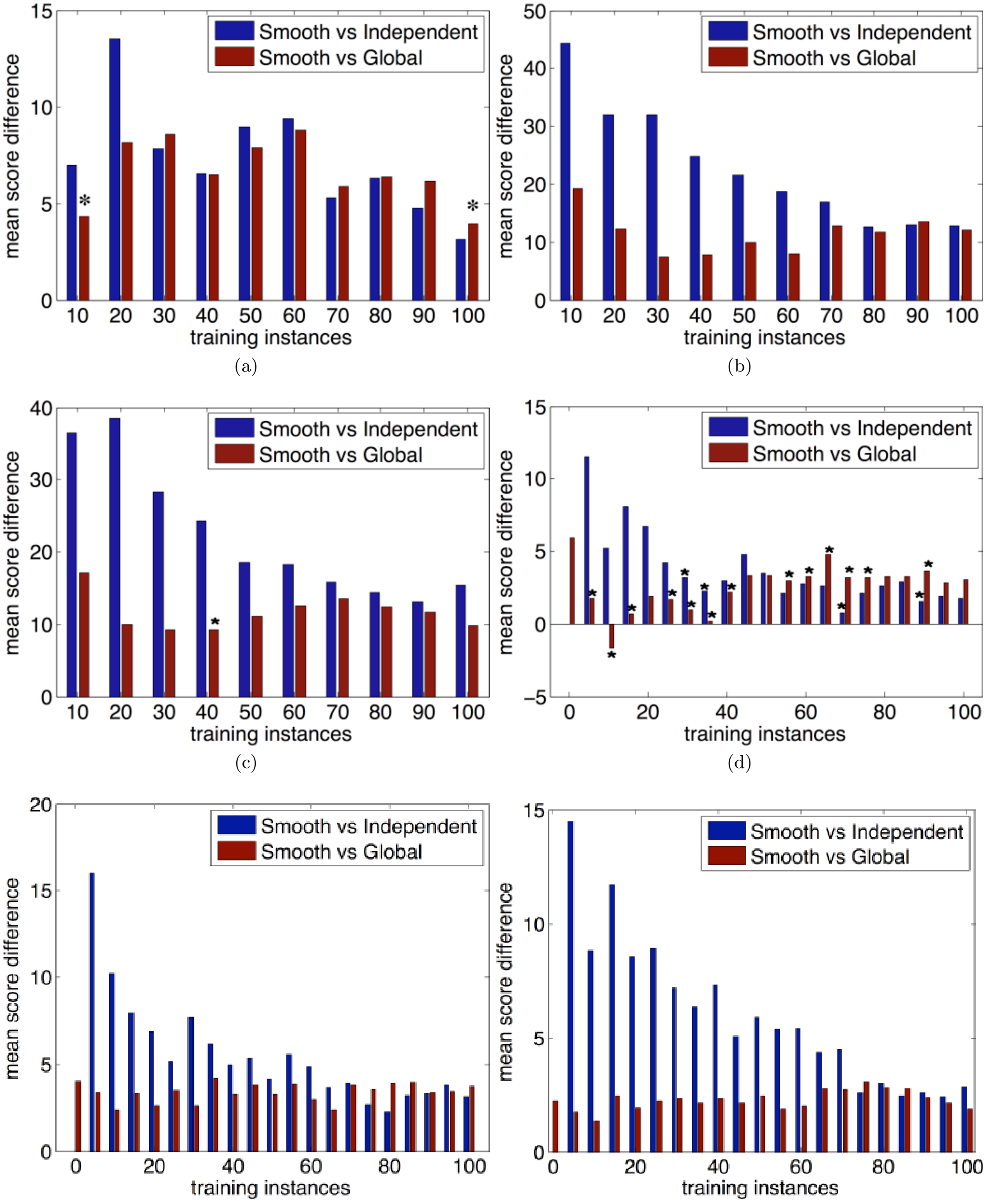
Figure 4: Improvement in cross-validation score of **Smooth** versus **Multitask** as training size increases for one condition (blue), two conditions (green) and three conditions (red).

7.3 COMPARISON AGAINST TRUE STRUCTURES

7.3.1 Edit Distance

The true structures that generated the synthetic data are known, therefore we can directly compare learned models to the truth. We measure the edit distance (the number of edges that must be added, deleted or reversed to change from one graph to the other) between each algorithm's learned structures and the truth.

We compare edit distance for structures learned from one, two and three conditions using the **Smooth**, **Independent**, and **Global** alogrithms. The difference in edit distance between algorithms is often insignificant (Figure 5). This is likely due to the small size of the Bayesian networks being learned. However, as



(a)

(b)

(c)

(d)

(e)

(f)

Figure 3: Improvement in cross-validation score of **Smooth** versus **Independent** (blue) or **Global** (red) as training size of every context increases for one condition (a), two conditions (b) and three conditions (c). Improvement in cross-validation score of **Smooth** versus **Independent** (blue) or **Global** (red) as training size increases for a particular context (other contexts have 1000 samples), for one condition (d), two conditions (e) and three conditions (f). * indicates that the paired T test is not significant (p=0.5).

the previous results show, small differences in structure can mean big differences in generalizing to test data. As the size of the training set increases, differences start to appear when there are two or three conditions. **Smooth** learns increasingly better structures when compared to **Global**, while **Independent** does increasingly better than **Smooth**. As the amount of training data increases, it becomes less important to leverage data from other contexts, but **Smooth** still performs reasonably well due to its flexibility to leverage some data and allow for differences in structure.

7.4 PBAIC RESULTS

7.4.1 Prediction

The goal of the PBAIC is to predict what is happening in a movie based on the fMRI data of the subject watching the movie (Schneider and Siegle, 2006). To make predictions, we learn a Bayesian network for each context, then score each test sample against each of the four learned structures. We choose the context with the highest score as our prediction. Structures for each context are learned either independently, **Indpendent**, or smoothing with our task-relatedness Hamming distance metric, **Smooth**. Accuracy (number of correct predictions divided by the total number of test samples) is measured for each context for each learning algorithm.

Note that this test is performed primarily as a validation of our method of learning structures and the construction of the task-relatedness metagraph. We are really interested in learning the interactions of ROIs in the brain, not in predicting behavior. This is inherently an unsupervised learning problem as the functional structure of the brain is unknown.

As Table 1 shows, the prediciton accuracy of **Smooth** is not good, but it is no worse than learning each Bayes net independently (**Independent**). This is interesting because it shows that we are not losing discriminative information by leveraging data across tasks.

Table 1: PBAIC Prediction Accuracy Results. Results are given in percent of total test samples.

Context	Training Samples	**Smooth**	**Indep-endent**
00	345	24.66 ± 2.05	24.24 ± 1.88
01	40	23.49 ± 7.06	20.70 ± 4.71
10	356	24.93 ± 2.32	25.57 ± 3.24
11	127	25.26 ± 3.43	24.38 ± 3.48

7.4.2 Cross Validation

Another validation of our learned structures tests the generalizability of the structure against holdout data. In this experiment, we score learned structure against the data from the same subject watching the second movie. Of particular importance is learning structures for contexts in which there is little data. Table 2 shows the amount of available data for training from each context. We actually learn our structures from half this data, using a random sub-sampling of 50% of the data.

Results of the paired T-test showing when our algorithm **Smooth** outperforms the other alogrithms are presented in Table 2. All algorithms are compared against **Smooth**. ”Yes” indicates that **Smooth** had better scores on the holdout set. ”No” indicates that **Smooth** had worse scores. * indicates insignificance in score difference according to the paired T-test (p=0.05).

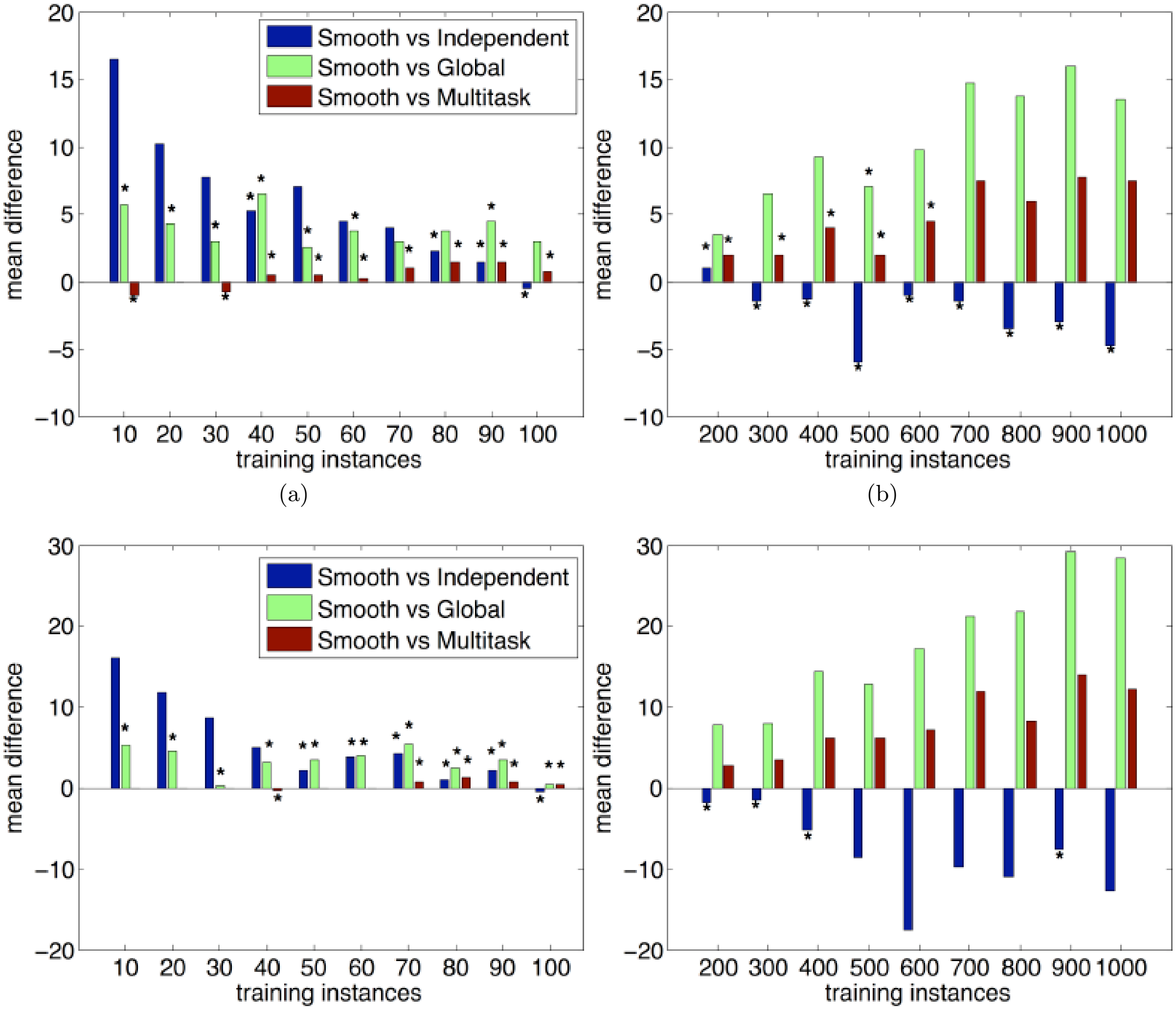
We always perform better than **Multitask** and never perform worse than **Global**. Hence, learning different structures for each context is important. Because **Global** and **Multitask** group more data together, they perform worse. The comparison against **Independent** is a bit less clear. In two cases, there is no significant difference. However, when there is very little training data, we perform better and when there is a lot of training data, we perform worse. When there is little data, **Independent** does not have enough information to learn the quality of structure that **Smooth** learns.

Table 2: PBAIC Cross Validation Results. ”Yes” indicates that **Smooth** has better scores on holdout data. ”No” indicates that **Smooth** has worse scores on holdout data. * indicates that there is not a significant difference in scores.

Context	Training Samples	**Indep-endent**	**Global**	**Multi-task**
00	345	*	Yes	Yes
01	40	Yes	Yes	Yes
10	356	No	Yes	Yes
11	127	*	*	Yes

8 CONCLUSIONS

Both the synthetic data and the fMRI data show that it is advantageous to leverage data in a manner consistent with the relatedness of tasks when learning Bayesian networks. The generalizability of the these structures in particular is strengthened. Our algorithm also provides a mechanism for allowing individuality in structures for specific contexts. We have shown that these two properties are particularly important as the number of conditions increases creating more complex relationships between contexts.



(a)

(b)

(c)

(d)

Figure 5: Improvement in edit distance from ground truth for **Smooth** versus **Independent** (blue), **Global** (green), or **Multitask** (red) as training size of every context increases for two conditions (a,b) and three conditions (c,d). Results for one condtion omitted because nearly every datapoint is insignificant. Positive values indicate that **Smooth** is closer to ground truth than the other algorithm (distance is negated to keep graphs consistent with "up is good" convention). Negative values indicate that the other algorithm is closer to ground truth than **Smooth**. * indicates that the paired T test is not significant (p=0.5).

References

R. Caruana. Multitask learning. *Machine Learning*, 28(1): 41–75, 1997.

E. Eaton, M. Desjardins, and T. Lane. Modeling transfer relationships between learning tasks for improved inductive transfer. In *ECML PKDD '08*, pages 317–332, Berlin, Heidelberg, 2008. Springer-Verlag.

D. Heckerman, D. Geiger, and D. M. Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, September 1995.

R. Luis, L. Sucar, and E. Morales. Inductive transfer for learning bayesian networks. *Machine Learning*, December 2009.

M. Nassar, R. Abdallah, H. A. Zeineddine, E. Yaacoub, and Z. Dawy. A new multitask learning method for multiorganism gene network estimation. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 2287–2291, July 2008.

A. Niculescu-Mizil and R. Caruana. Inductive transfer for bayesian network structure learning. In *AISTATS 2007*, volume 2, pages 339–346, 2007.

W. Schneider and G. Siegle. Pittsburgh brain activity interpretation competition guidebook, http://www.ebc.pitt.edu/2006/competition.html, 2006.

B. Yackley, E. Corona, and T. Lane. Bayesian network score approximation using a metagraph kernel. In *NIPS*, pages 1833–1840. MIT Press, 2008.