

Exploiting Parameter Related Domain Knowledge for Learning in Graphical Models

Radu S. Niculescu and Tom M. Mitchell
Dept of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
{stefann,tom.mitchell}@cs.cmu.edu

R. Bharat Rao
Clinical CAD
Siemens Medical Solutions
Malvern, PA 19355
bharat.rao@siemens.com

Abstract

Building accurate models from a small amount of available training data can sometimes prove to be a great challenge. Expert domain knowledge can often be used to alleviate this burden. Parameter Sharing is one such important form of domain knowledge. Graphical models like HMMs, DBNs and Module Networks use different forms of Parameter Sharing to reduce the variance in the parameter estimates. The goal of this paper is to present a theoretical approach for learning in presence of several other types of Parameter Related Domain Knowledge that go beyond the ones in the above models. First, we introduce a General Parameter Sharing Framework that describes the models just mentioned, but allows for much finer grained parameter sharing assumptions. In this framework, we present sound procedures for parameter learning from both a Frequentist and a Bayesian point of view, from both complete and incomplete data, in the case where a domain expert specifies in advance the structure of the graphical model, and the subsets of parameters to be shared. Second, we describe a hierarchical extension of this framework based on Parameter Sharing Trees. Finally we present algorithms for using domain knowledge that specifies that certain groups of parameters share certain properties. In particular, we consider two kinds of constraints: first kind states certain groups of parameters share the same aggregate probability mass and second kind states the ratio of the parameters is preserved (shared) in several groups. As an example, we derive a novel form of parameter sharing for Bayesian Multinetworks.

1 Introduction

The task of learning models for many real-world problems requires researchers to incorporate problem Domain Knowledge into the learning algorithm – there is rarely enough training data to enable the learning of the structures and underlying relationships in the problem. Domain Knowledge comes in many forms: Domain Knowledge about relevance of variables, also called Fea-

ture Selection, can help us ignore certain variables when building our model. Domain Knowledge specifying conditional independencies among variables can guide our search over possible model structures. Both these forms have been extensively studied.

This paper presents a theoretical approach for incorporating a different kind of knowledge into learning: Domain Knowledge about relationships among parameters, specifically knowledge about sharing properties of groups of parameters across several conditional probability distributions (CPDs) in graphical models parameterized by conditional probability tables (CPTs). It is somewhat obvious that leveraging such information can greatly benefit learning, as proved by graphical models like HMMs, DBNs and Module Networks. For instance, if a parameter is shared across multiple subproblems, the variance in the estimates can be reduced by learning from all the relevant data from the subproblems, as opposed to estimating the parameter independently for each subproblem.

In Section 3, we describe a *General Parameter Sharing Framework* for making very fine-grained parameter sharing assumptions that go beyond the ones specified in the above mentioned models, and present sound algorithms for learning, in many different scenarios. In this framework, the CPDs in the model are partitioned and a parameter can either be shared among all CPDs in a subset or not shared at all. Some formal guarantees are provided about the derived estimators. We also discuss a scoring metric for comparing different sets of sharing assumptions and suggest how this metric can be used for automatically recovering relevant parameter sharing, provided that the expert does specify only a limited amount of domain knowledge. Section 4 describes a hierarchical extension of the previous framework, where we allow some parameters to be shared across all CPDs, then partition recursively the set of CPDs, allowing new parameters to be shared among subsets (of CPDs) of the

partition. In section 5 we discuss further extensions that investigate sharing properties of groups of more than one parameter across several CPDs. In particular, we consider two kinds of constraints: first kind states certain groups (each group belonging to a different CPD) of parameters share the same aggregate probability mass and second kind states the ratio of the parameters is preserved (shared) in several such groups. Section 6 shows an example that demonstrates that graphical models learned using parameter sharing, are preferred to the alternatives of sharing no parameters or using a global model that effectively shares all parameters on a task of email modelling using Bayesian Multinetworks. We plan to apply our work to other more complex real-world problems in the near future. The final section summarizes the main contributions of this paper and suggests some directions for future work. We begin by briefly reviewing some related research.

2 Related Work

The standard way of representing parameter related domain knowledge in graphical models is by using Dirichlet priors. It can be proved [7], that under certain conditions, choosing Dirichlet priors over the parameters of a Bayesian Network is inevitable. However, Dirichlet priors are specified over a CPD(Conditional Probability Distribution) in a CPT (Conditional Probability Table) and therefore do not represent constraints among parameters in different CPDs. An approach of relaxing the local independence assumption for binary variables is investigated in [8]. In [13], the authors compare several other smoothing methods for learning language models.

Parameter sharing has also been explored in a variety of graphical models. For example, HMMs [10] and Dynamic Bayes Nets [9] make the assumption that the CPTs (Conditional Probability Tables) corresponding to a given variable are the same at each point in time. In Probabilistic Relational Models [5], objects of a certain type class share the way they depend on related objects of other type classes. In Module Networks [12], variables in the same module exhibit similar behavior i.e. they have the same set of parents, same set of values and probabilistically they depend on their parents in the same way and therefore their corresponding CPTs can be learned together. Context Specific Independence (CSI) [1] states conditional independencies that hold only in certain contexts i.e. of the form $P[X|Y, Z, C = c] = P[X|Z, C = c]$ and therefore can specify that some CPDs in the CPT for a variable X are the same (they share all parameters in those CPDs). CSI can be exploited to efficiently encode and learn CPTs using default tables, decision trees and decision graphs as described in [3] and [4].

As an example, in section 6 we will illustrate the utility of our General Parameter Sharing Framework by showing how it allows a novel kind of parameter sharing in Bayesian Multinets [2],[6] for the task of modelling emails coming from several users.

3 A General Parameter Sharing Framework

We present a General Parameter Sharing Framework that describes learning in a broad category of graphical models. We show how to approach learning within this framework from both a Frequentist and a Bayesian point of view, from both complete and incomplete data, in the case where a domain expert specifies in advance the structure of the graphical model, and an arbitrary subset of parameters to be shared.

Each CPD (Conditional Probability Distribution) represents the constraint that its parameters should sum up to one. We denote by C the set of all CPDs in all CPT's in the graphical model. Below we will describe two assumptions that suffice to derive learning procedures that take advantage of Parameter Sharing:

Parameter Sharing. The CPD set C can be partitioned as $C = \cup_{k=1}^m C_k$, such that some parameters (denote their set by G_k) are shared (appear exactly once in each of the different CPDs) within the set C_k , but not shared within the same CPD or with other sets of CPDs of the partition. Let $L_k = C_k \setminus G_k$ be the set of local (not shared) parameters.

Decomposability of Log-Likelihood. For any complete dataset D , the log-likelihood can be decomposed as:

$$\log(P(D|\theta)) = \sum_{k=1}^m h_k(C_k)$$

$$\text{where } h_k(C_k) = \sum_{\theta_{g_k} \in G_k} N_{gk} \cdot \log \theta_{gk} + \sum_{\theta_{lck} \in L_k, c \in C_k} N_{lck} \cdot \log \theta_{lck}$$

Above, N_{gk} represents the cumulative count corresponding to shared parameter θ_{gk} (appears in each CPD in C_k exactly once) and N_{lck} is the count corresponding to local parameter θ_{lck} (in CPD $c \in C_k$).

We now discuss several graphical models which all fit within our framework and satisfy the *Parameter Sharing* assumption. For instance, in HMMs and Dynamic Bayes Nets, the same variable has the same CPT at different time instants. Therefore, a subset C_k of the partition is made out of the CPDs in the CPTs which correspond to a given variable X and the same instantiation of the parents $PA(X) = pa$ across all time instants. In Module Networks, all variables in the same module share the same set of parents and have the same CPTs. Consequently, the subsets of the partition

contain the set of CPDs corresponding to all variables in a module for a given instantiation of the parents of those variables. Context Specific Independence is used to specify conditional independencies that hold in certain contexts and therefore is useful to specify which CPDs should be equal in a CPT for a fixed random variable. In this case, the subsets of the partition consist of CPDs in the same CPT which are assumed equal because of the CSI assumptions. However, note that our framework allows for much more flexibility in parameter sharing. We can share at the level of each parameter, not just at the whole CPD or table level. Also, the CPDs in the same sets of partition do not have to be the same size. Moreover, a shared parameter doesn't need to be in the same position in different CPDs within the same set of the partition.

Below we discuss different scenarios in our framework and provide learning algorithms.

3.1 Learning - Frequentist Approach, Full Data Observability, Known Structure. A Frequentist tries to learn one single model that "best" fits the data. When structure is known in advance, this often translates into finding the Maximum Likelihood Estimators (MLEs) for the parameters in the model. Subsequently, we are also going to discuss Maximum A Posteriori (MAP) Estimators. The MLEs in our General Parameter Sharing Framework, when structure and parameter sharing is specified by a domain expert, can be computed using the following theorem:

THEOREM 3.1. *Assume we are given a graphical model with known structure that satisfies the Parameter Sharing and Decomposability of Log-Likelihood assumptions. Then, the Maximum Likelihood Estimators (MLEs) for the parameters in our graphical model are given by:*

$$\hat{\theta}_{gk} = \frac{N_{gk}}{\sum_{\theta_{g'k} \in G_k} N_{g'k} + \sum_{\theta_{lck} \in L_k} N_{lck}}$$

$$\hat{\theta}_{lck} = \frac{\sum_{\theta_{l'c'k} \in L_k} N_{l'c'k}}{\sum_{\theta_{g'k} \in G_k} N_{g'k} + \sum_{\theta_{l'c'k} \in L_k} N_{l'c'k}} \cdot \frac{N_{lck}}{\sum_{\theta_{lck} \in L_k} N_{lck}}$$

The MLEs for shared parameters look similar to the ones in the case of standard Bayes Nets. However, the MLEs for local parameters are a product of two factors. First factor represents the probability mass that remains after subtracting the shared parameters. The second factor basically says that this remaining "local" probability mass in a CPD is split into values proportional to the counts corresponding to the local parameters in that CPD.

Proof of Theorem 3.1 (Sketch): Because of the *Parameter Sharing* and *Decomposability of Log-Likelihood* assumptions, the problem of maximizing the data likelihood can be broken down into a set of independent optimization subproblems:

$$P_k : \operatorname{argmax}_{C_k} \{h_k(C_k) \mid g_{ck}(C_k) = 0, \forall c \in C_k\}$$

where

$$g_{ck}(C_k) = (\sum_{\theta_{gk} \in G_k} \theta_{gk}) + (\sum_{\theta_{lck} \in L_k} \theta_{lck}) - 1 = 0$$

When all counts are positive, it can be easily proved that P_k has a global maximum which is achieved in the interior of the region determined by the constraints. In this case the solution of P_k can be found using Lagrange Multipliers. Introduce Lagrange Multipliers $\lambda_k = (\lambda_{ck})_{c \in C_k}$ for each CPD in C_k . Let $LM(C_k, \lambda_k) = h_k(C_k) - \sum_{c \in C_k} \lambda_{ck} \cdot g_{ck}(C_k)$. Then the point which maximizes P_k is among the solutions of the system $\nabla LM(C_k, \lambda_k) = 0$. It turns out that this system has a unique solution which is in fact the one in the statement of the theorem. Since the function achieves its maximum in the interior of the constraint region, it means that the solution of the system supplies the MLEs. In the case when some of the counts are zero, the corresponding parameters don't even appear in the likelihood function and the optimization problem then has inequality constraints but eventually the MLEs are given by the same formulas. This concludes the sketch of our proof. \square

Note that, if the expert makes a mistake about the structure or about the shared parameters of the model, then the learned distribution may be much different from the true distribution of the data. Below we will provide formal guarantees about our estimators.

In order to present these results, let us introduce the notion of *True Probabilistic Counts (TPC)*. Suppose P is the true distribution of which data is sampled. If θ_{lck} is the local parameter of the graphical model that is supposed to describe $P(X = x \mid PA(X) = pa)$, then let $TPC_{lck} = P(X = x, PA(X) = pa)$. If θ_{gk} is the global parameter of the graphical model that is supposed to describe the set $\{P(X_1 = x_1 \mid PA(X_1) = pa_1), \dots, P(X_s = x_s \mid PA(X_s) = pa_s)\}$, let $TPC_{gk} = \sum_{i=1}^s P(X_i = x_i, PA(X_i) = pa_i)$. Let P^* be the distribution that factorizes according to the structure provided by the expert and has parameters given by theorem 3.1 where the counts are replaced by the *True Probabilistic Counts*.

THEOREM 3.2. *P^* is the closest distribution to P (in terms of $KL(P, \cdot)$) that factorizes according to the given structure and obeys the expert's parameter sharing assumptions.*

Proof of Theorem 3.2 (Sketch): Let Q be such a distribution. Minimizing $K(P, Q)$ is equivalent to maximizing $\sum_d P(d) \cdot \log Q(d)$. Let θ be the set of parameters that describe this distribution Q . After breaking the logarithms into products of logarithms based on the factorization given by the provided structure, our optimization problem reduces to the maximization of $\sum TPC_{gk} \cdot \log \theta_{gk} + \sum TPC_{lck} \cdot \log \theta_{lck}$. The solution of this problem is obviously given by theorem 3.1. This is equivalent to the fact that P^* (see the definition above) minimizes $KL(P, \cdot)$ out of all the distributions that factorize according to the given structure and obey the expert's sharing assumptions. \square

THEOREM 3.3. *With infinite amount of data, the distribution \hat{P} given by the ML Estimators in Theorem 3.1 converges to P^* with probability 1.*

Proof of Theorem 3.3 (Sketch): Assume the number of data points in a dataset sampled from P is denoted by n . According to the Law of Large Numbers, we have $\lim_{n \rightarrow \infty} \frac{N_{lck}}{n} = TPC_{lck}$ and $\lim_{n \rightarrow \infty} \frac{N_{gk}}{n} = TPC_{gk}$ with probability 1. This is equivalent to the fact the \hat{P} converges to P^* with probability 1. \square

3.2 Learning - Frequentist Approach, Partial Data Observability, Known Structure. If some of the attributes of the data points are not observed, then the counts N_{gk} and N_{lck} should be treated as random variables (random counts). One still can do Maximum Likelihood parameter estimation using the 2-step EM algorithm. Given the formula of the log-likelihood for complete data and the parameter sharing assumptions, by completing the data in all possible ways, it is easy to see that the standard EM algorithm applied to our setting will repeat the following steps until convergence is reached:

E-Step: Use any inference algorithm to compute expected counts under the current parameter estimates $\hat{\theta}$. If just starting, assign $\hat{\theta}$ randomly or according to some domain knowledge.

M-Step: Reestimate the parameters by maximizing the data likelihood using Theorem 3.1, assuming that the observed counts are equal to the expected counts given by the E Step.

3.3 Learning - Bayesian Approach, Full Data Observability, Known Structure. From a Bayesian point of view, each choice of parameters is possible, but some choices have higher probability of occurring. Therefore, to do model averaging, we need to specify priors over the space of parameters. It can

be proved [7], that under certain conditions, choosing Dirichlet Priors over the parameters of a Bayesian Network is inevitable. Two of the assumptions that are usually made are local and global independence. Because of the *Parameter Sharing* assumption, the local and global independence assumptions may not hold. However, in our case we can make the following similar assumption:

Subset Independence Assumption: parameters in different subsets of the partition are independent of each other. In other words, $p(\theta) = p(C) = \prod_{k=1}^m p(C_k)$.

Under this assumption it is enough to define a prior over the parameters in each subset of the partition. A *Subset Specific Dirichlet Distribution* over C_k of parameters $\{\alpha_{gk} | gk \in G_k\} \cup \{\alpha_{lck} | lck \in L_k\}$ will have the following formula:

$$p(C_k) = \frac{1}{Z_k} \cdot \prod_{\theta_{gk} \in G_k} \theta_{gk}^{\alpha_{gk}-1} \cdot \prod_{\theta_{lck} \in L_k, c \in C_k} \theta_{lck}^{\alpha_{lck}-1}$$

$$SP_k = \left\{ \sum_{\theta_{gk} \in G_k} \theta_{gk} + \sum_{\theta_{lck} \in L_k} \theta_{lck} = 1 \quad \forall c \in C_k \right\}$$

over the space

Above, Z_k is a normalization constant which can be found by enforcing the condition that the integral of the pdf should be equal to one:

$$Z_k = \prod_{gk \in G_k} \Gamma(\alpha_{gk}) \cdot \prod_{c \in C_k} \frac{\prod_{lck \in L_k} \Gamma(\alpha_{lck})}{\Gamma(\sum_{lck \in L_k} \alpha_{lck})} \cdot \frac{\Gamma(\sum_{lck \in L_k, c \in C_k} \alpha_{lck} - |C_k| + 1)}{\Gamma(\sum_{lck \in L_k, c \in C_k} \alpha_{lck} + \sum_{gk \in G_k} \alpha_{gk} - |C_k| + 1)}$$

There are several interesting properties of this Subset Specific Dirichlet Distribution. First, the joint probability distribution over the shared parameters is a standard Dirichlet. Second, with no parameter sharing, this distribution is a product of independent standard Dirichlet distributions, one for each CPD in C_k . However, if there are both shared and local parameters, then the joint probability (obtained by marginalization) over a CPD $c \in C_k$ is not a standard Dirichlet. Finally, because of the *Decomposability of Log-Likelihood*, it is easy to see that the posterior $p(\theta|D) \propto p(D|\theta) \cdot p(\theta)$ is also a product of Subset Specific Dirichlet distributions and therefore the collection of multinomials with shared parameters (over C_k) and the Subset Specific Dirichlet distribution are conjugate distributions.

Now that we have defined priors over the space of parameters, it is trivial to compute MAP Estimators.

Because of *Decomposability of Log-Likelihood*, computing MAP Estimators is just a matter of adding corresponding Subset Specific Dirichlet exponents (not parameters) to the observed counts in the ML Estimators in (1) and (2).

From a Bayesian point of view, we are interested in predicting the next data point given previous data points. This can be written as follows:

$$p(D_{n+1}|D_1, \dots, D_n) = \frac{\int_{\cup_{SP_k} p(D_{n+1}, \dots, D_1|\theta) \cdot p(\theta) d\theta}{\int_{\cup_{SP_k} p(D_n, \dots, D_1|\theta) \cdot p(\theta) d\theta}$$

As stated above, $p(\theta|D) \propto p(D|\theta) \cdot p(\theta)$ is also a product of Subset Specific Dirichlet distributions. Therefore both integrals in (4) are products of normalization constants for some Subset Specific Dependent Dirichlet distributions. We already showed that we can compute these normalization constants and thus we can easily compute these integrals. Looking again at the formula for Z_k , it is worth mentioning that most of the factors will cancel out when computing the ratio of the two integrals. The only ones remaining would be the ones where the introduction of the new data point D_{n+1} would increment the counts.

3.4 Learning - Bayesian Approach, Partial Data Observability, Known Structure. When data is incomplete, we can no longer write $p(D|\theta)$ as a product of Subset Specific Dirichlet Distributions as we did in the complete data case discussed above. In this case, let U be the set of missing values such that $D \cup U$ is a complete dataset. Then, $\int p(D|\theta) \cdot p(\theta) d\theta = \sum_U \int p(D, U|\theta) \cdot p(\theta) d\theta$. Therefore, in the case of incomplete data, $\int p(D|\theta) \cdot p(\theta) d\theta$ is a sum of products of normalization constants for certain Subset Specific Dirichlet Distributions and can be used to compute the ratio in (4).

The above procedure for performing Bayesian Estimation is computationally expensive because the number of terms in the summation grows extremely fast. If only one binary value is missing in each of the n examples, there will be 2^n terms in the summation. Approximation techniques for $p(D|\theta)$ when data is incomplete are available for standard Bayes Nets, but investigating them in our parameter sharing framework is beyond the scope of this paper.

3.5 Comparing Parameter Sharing Schemes.

A *Parameter Sharing Scheme PShS* over a graphical model structure S is a set of valid parameter sharing assumptions (of the type specified in the general parameter sharing framework) on top of structure S . We

remind the reader that in all our work, the structure of the model is given by an expert and does not change. Learning structure in presence of parameter sharing is subject for future work.

A *PShS* helps reduce the variance in parameters' estimates. In section 6, we empirically show that this translates in estimated distributions closer in KL distance to the true underlying distribution than the ones estimated without taking advantage of parameter sharing. There we also show that the more valid parameter sharing assumptions we know, the better the estimates. Therefore it is important to recover these assumptions, even when the expert doesn't know the parameter sharing or can specify only a limited set of such assumptions.

Assume that a dataset D of examples is provided and our goal is to learn an *optimal PShS* over a given structure S . In order to do this, we need to be able to compare different PShSs. We propose a metric similar to the one used for structure search i.e. we try to find the *PShS* that maximizes $P(D|PShS)$ (*Sharing Score*). Averaging over all sets of parameters θ consistent with *PShS*, we obtain:

$$p(D|PShS) = \int_{\theta \in PShS} p(D|\theta, PShS) \cdot p(\theta|PShS) d\theta$$

It is easy to notice that the quantity inside the integral represents a product of Subset Specific Dirichlet Distributions and therefore $P(D|PShS)$ is a product of normalization constants for such distributions. We previously showed how such constants can be computed in the case of complete data. Therefore, in the case of complete data, it is straightforward how to obtain the *Sharing Score*.

What happens in the case of incomplete data is a little bit more complicated. In this situation we can write: $p(D|\theta) = \sum_{D' \text{ complete, consistent with } D} p(D'|\theta)$. Therefore, $P(D|PShS)$ will be a sum of products of normalization constants for Subset Specific Dirichlet Distributions. It is worth pointing out that the number of such products grows at least exponentially with the amount of missing data. However, one can think of techniques to approximate $P(D|\theta)$ with Subset Specific Dirichlet Distributions by processing data points one at a time and updating the current parameter prior accordingly. In this case, the order in which examples are processed matters. Investigating this is however beyond the scope of this paper.

We suggest that the above scoring metric can be used to uncover the underlying *PShS* via some hill climbing techniques i.e. one can think of deriving the current *PShS* candidate from previous one by using small modifications. Also, restricting the set

of potential *PShS* can be useful. For example, in module networks one can restrict the variables that can belong together in a module or in HMMs one would specify that one expects only transition tables to have sharing, but no sharing between transition tables and the tables describing the observed output. In addition, our suggested method can be useful when combined with an expert who knows a subset of sharing assumptions because this restricts further the superset of valid parameter sharing schemes.

4 Hierarchical Parameter Sharing Framework

Here we present a hierarchical extension of the framework in the previous section. This will address some of the limitations of the constraints that could be incorporated in the parameter sharing framework described before. In order to derive our main results in this section, we first need to make an important assumption about the graphical models we are working on:

Log-Likelihood Assumption. For any complete dataset of examples D , the log-likelihood can be written as: $\log(P(D|\theta)) = \sum_{\theta_i} N_{\theta_i} \cdot \log \theta_i$ where N_{θ_i} represents the cumulative observed count for parameter θ_i (which may appear in multiple places in the graphical model).

We are now ready to describe our learning framework. First of all, we present Parameter Sharing Trees as a way to encode hierarchical parameter sharing assumptions and second we show how one can take advantage of such a Parameter Sharing Tree in order to alleviate learning. We are going to discuss only the derivation of ML and MAP Estimates from complete data. Same as in the previous section, the EM adaptation for learning from incomplete data is just a matter of estimating the expected counts under current parameter estimates using any available inference algorithm, then reestimate the parameters using the expected counts as if they were observed counts. Also, the Bayesian approach for learning from complete and from incomplete data goes along the same lines with the discussion in previous section.

4.1 Parameter Sharing Trees. Again, let C represent the set of all CPDs in the graphical model. Each such CPD introduces a constraint on the possible values that the parameters θ can take. A *Parameter Sharing Tree (PST)* is a tree with the following properties:

- Each node v of the tree consists of a pair $(Scope(v), Shared(v))$, where $Scope(v)$ is a subset of C and $Shared(v)$ represents a non-empty set of parameters that are shared across these CPDs.

A parameter from $Shared(v)$ is a parameter that is known to appear exactly once in each of the CPDs in $Scope(v)$, but it is not shared multiple times within one CPD, nor with CPDs outside the $Scope$.

- By convention, $Scope(Root) = C$ (this amounts to the fact that we would like to allow for the extreme situation when a parameter is shared by all CPDs in the graphical model).
- The *Scopes* of the direct descendants of a node v form a partition of $Scope(v)$. Therefore, the PST will describe a recursive way of partitioning C , with the leaf level being the finest grain of such partition.
- A parameter cannot be shared in multiple places (different nodes of the tree). Because of the recursive partitioning of the CPDs, this amounts to the fact that $Shared(v)$ is disjoint with all *Shares* on the path from v to the root of *PST*.
- Each parameter θ of the graphical model is shared exactly once i.e. there exists a node v such that $\theta \in Shared(v)$. One may argue that there are parameters which are not shared at all, but for all nodes v that have CPDs in their *Scope* such that there remain unshared parameters, one can partition those nodes further in leaves that have only one CPD in their *Scope*, for which previously unshared parameters become shared at the level of that single CPD.

Let $Desc(v)$ be the set of descendants of node v (included) in and $Shared(Desc(v)) = \bigcup_{v' \in Desc(v)} Shared(v')$. Also, denote by $Ancestors(v)$ the set of nodes on the path from v to the root of the tree and $Shared(Ancestors(v)) = \bigcup_{v' \in Ancestors(v)} Shared(v')$.

4.2 Learning - Frequentist Approach, Full Data Observability, Known Structure. Assume we are given a graphical model with known structure that satisfies a set of parameter sharing assumptions given by a *PST* T and also satisfies the *Log-Likelihood Assumption*. In this section we are going to present a theorem that will justify an algorithm for finding the Maximum Likelihood Estimators for the parameters in such a graphical model. Denote by $\hat{\theta}$ the Maximum Likelihood Estimators.

THEOREM 4.1. *Let v be a node of T and $\theta_i \in Shared(v)$. The following equality holds:*

$$\hat{\theta}_i = \left(1 - \sum_{\theta_j \in Shared(Ancestors(v))} \hat{\theta}_j \right) \cdot$$

$$\frac{N_{\theta_i}}{\sum_{\theta_k \in \text{Shared}(\text{Desc}(v))} N_{\theta_k}}$$

Proof of Theorem 4.1 (Sketch): By definition,

$$\hat{\theta} = \underset{\theta}{\text{argmax}} \left\{ \sum_{\theta_i} N_{\theta_i} \cdot \log \theta_i \mid \theta \text{ satisfies } T \right\}$$

Each CPD represents a constraint on the space of parameters: $g_c(\theta) = (\sum_{\theta_j \in c} \theta_j) - 1 = 0$. Because of parameter sharing, these constraints can involve some common variables. It is easy to show that, if all the cumulative counts are positive, the likelihood function has a global maximum inside the region determined by the constraints in T . In the case when there exist counts equal to 0, it is also easy to show that the ML estimators for the corresponding parameters are also zero. When the maximum is reached in the interior of the domain, one can apply Lagrange Multipliers to optimize for P_{ik} . Therefore, let us introduce new variables λ_c for each constraint (CPD) $c \in C$. The new function to optimize will be: $LM(\theta, \lambda) = \sum_{\theta_i} N_{\theta_i} \cdot \log \theta_i - \sum_c \lambda_c \cdot g_c(\theta)$. According to Lagrange Multipliers theory, any point that is a local maximum or minimum for the initial optimization problem and it is NOT on the border of the region defined by the constraints will be obtained as a (partial) solution of the system of equations:

$$\nabla LM(\theta, \lambda) = 0$$

Therefore $\hat{\theta}$ verifies the above system for some values of λ . Because $\frac{\partial L(\theta|D)}{\partial \theta_i} = \frac{N_{\theta_i}}{\theta_i}$ and $\frac{\partial g_c}{\partial \theta_i} = \lambda_c$ if CPD c contains θ_i (otherwise the partial derivative is zero), we get:

$$(4.1) \quad \hat{\theta}_i = \frac{N_{\theta_i}}{\sum_{c \in \text{Scope}(v)} \lambda_c} \quad \forall \theta_i \in \text{Shared}(v)$$

Let $S(v) = \sum_{\theta_j \in \text{Shared}(\text{Desc}(v))} \hat{\theta}_j$. We will prove by induction the following stronger claim:

$$S(v) = \frac{\sum_{\theta_j \in \text{Shared}(\text{Desc}(v))} N_{\theta_j}}{\sum_{c \in \text{Scope}(v)} \lambda_c}$$

and

$$\hat{\theta}_i = \left(1 - \sum_{\theta_j \in \text{Shared}(\text{Ancestors}(v))} \hat{\theta}_j \right) \cdot \frac{N_{\theta_i}}{\sum_{\theta_k \in \text{Shared}(\text{Desc}(v))} N_{\theta_k}}$$

Base case: If v is a leaf, then the distributions in $\text{Scope}(v)$ are equal. First part of the claim is verified directly from 4.1 and the second part follows because of the fact that the probabilities that add up to $S(v)$ are proportional to their corresponding counts.

Induction Step: Assume v is not a leaf and has direct descendants d_1, \dots, d_k for which the claim holds. It is obvious that $S(d_1) = \dots = S(d_k)$. Now, using the induction hypothesis, we obtain $S(d_1) = \dots = S(d_k) = \frac{\sum_{l=1}^k \sum_{\theta_j \in \text{Shared}(\text{Desc}(d_l))} N_{\theta_j}}{\sum_{c \in \text{Scope}(v)} \lambda_c}$. This combined with 4.1 gives us the first part of the claim. The second part of the claim now follows from 4.1 and the fact that

$$\frac{1}{\sum_{c \in \text{Scope}(v)} \lambda_c} = \frac{S(v)}{\sum_{\theta_j \in \text{Shared}(\text{Desc}(v))} N_{\theta_j}}$$

The proof of our theorem is now complete. \square

The above theorem yields an obvious recursive top-down, breadth-first algorithm to compute the ML Estimates of the parameters. The correctness of the algorithm is justified by theorem 4.1 and the fact that a node v is processed sometime after all the nodes on the path from v to the root are processed. The algorithm uses a queue Q to perform breadth-first traversal of the tree.

Algorithm:

STEP 1. Enqueue the root of the tree in Q .

STEP 2. If $Q = \emptyset$, STOP. Else, $v \leftarrow \text{Dequeue}(Q)$.

STEP 3. Compute $\hat{\theta}_i$ for all $\theta_i \in \text{Shared}(v)$.

STEP 4. Enqueue all children of v . GO TO STEP 2.

4.3 Learning - Bayesian Approach, Full Data Observability, Known Structure.

In order to be able to compute MAP estimates and predict an example from the previous ones by bayesian averaging, we have to define priors over the space of parameters. Note however that in our case, when hierarchical parameter sharing is present, the local and global parameter assumptions are violated. Next we will show how to define priors over the space of parameters that obey the constraints given by a *Parameter Sharing Tree* T . First of all, it would be nice if $P(\theta)$ and $P(D|\theta)$ are conjugate distributions. This suggests we chose our priors of the following form:

$$P(\theta) = \frac{1}{Z(T)} \prod \theta_i^{\alpha_i - 1}$$

Note that these priors are defined over the whole space of parameters and that a parameter θ_i can appear in multiple places in the graphical model (according to

the given *Parameter Sharing Tree*). In addition, the normalization constant depends heavily on the structure of the parameter sharing tree since T describes the constraints among parameters (the sum of parameters shared on the path from the root to any leaf should sum up to 1). If for finding MAP estimates the normalization constant is not important, it becomes important when computing $P(D_{n+1}|D_1, \dots, D_n)$ (as we saw in the previous section). Before showing how to compute $Z(T)$, let us define the *Generalized Dirichlet Integral* to be:

$$\int_{\sum \theta_i = S} \prod \theta_i^{\alpha_i - 1} d\theta = (1-S)^{(\sum \alpha_i) - 1} \cdot \int_{\sum \theta_i = 1} \prod \theta_i^{\alpha_i - 1} d\theta$$

The last integral is a normalization constant for a standard Dirichlet distribution and this gives us a way for computing the *Generalized Dirichlet Integral*. Now, the constant $Z(T) = \int_{\theta \text{ obeys } T} \prod \theta_i^{\alpha_i - 1} d\theta$ can be recursively computed as follows. First, note that this integral can be evaluated starting with the parameters from the leaf level. For each leaf v , the integral over the parameters involved in $Shared(v)$ is a *Generalized Dirichlet Integral*. The effect of computing this integral is to get the constant given by the Standard Dirichlet and propagate upwards a single parameter $(1 - S(v))$ with cumulative Dirichlet parameter $(\sum_{\theta_i \in Shared(v)} \alpha_i) - 1$. Now, it is easy to see that this parameter is the same for all leaves that belong to the same parent p . This will make the integral over the parameters in $Shared(p)$ and the new parameter to be also a *Generalized Dirichlet Integral* and the procedure continues as described above until we end the computation at the root level. This concludes or sketch of showing how one can recursively compute $Z(T)$ using *Generalized Dirichlet Integrals*. Once the normalization constant is computed, bayesian prediction can be done in exactly the same way as in the previous section.

5 Sharing Properties of Subsets of Parameters

In this section we present other extensions to the framework presented in section 3. Again, the set C of CPDs in the graphical model is partitioned as: $C = \cup_{k=1}^m C_k$, where the distributions in a subset C_k are going to be constrained together. Moreover, assume there are no constraints tying CPDs in different subsets of the partition. If in section 3 the constraints were individual parameter equalities, in this section we are studying constraints that involve whole subsets of parameters within the same CPD.

5.1 Probability Mass Sharing. Here we are going to show how to do Maximum Likelihood learning in the case when the aggregate probability mass of a certain

parameter type is the same across all CPDs in a given subset C_k . For example, we would like to show how to take advantage of constraints like: "The frequency of nouns in English is the same as the frequency of nouns in Spanish", when modelling the word probability in each of the two languages. In these case, types would be: nouns, verbs, etc.

Before stating the main result of this subsection, let us introduce few notations. Assume for a specific k , the parameters in C_k may have the following types: T_1, \dots, T_s . Denote by θ_i^j the i^{th} parameter in j^{th} CPD in C_k . Each parameter has exactly one type. For example, in the above example, $P(Computer|English)$ has type Noun, while $P(Blue|English)$ has type Adjective. We would like to stress the fact that in our framework, C_k is an arbitrary subset of CPDs in the graphical model; these CPDs can have different number of parameters and they can belong to different CPTs. Formally, the constraints that we are dealing with are given by:

Probability Mass Sharing Assumption. For all types T_l and for any j_1^{th} and j_2^{th} distributions in C_k , the following holds:

$$\sum_{\theta_i^{j_1} \in T_l} \theta_i^{j_1} = \sum_{\theta_i^{j_2} \in T_l} \theta_i^{j_2}$$

Back to our example, this translates into: "The aggregate probability of Nouns is the same in all modelled languages and same holds for other grammatical categories/types." It might seem a little restrictive to have each parameter belong to one type because, for instance, one may argue that maybe only the probability of Nouns is being shared across languages. However, even if one specifies the Probability Mass Sharing Assumption only for Nouns, the rest of parameters (non-nouns) verify the same constraint and therefore that is equivalent to introducing a new dummy type that contains every other parameter in C_k .

Since there are no constraints between CPDs in different C_k s, one can break the optimization for finding the ML Estimates into a set of independent optimization problems. The function to optimize for each C_k will be $f(C_k) = \sum N_i^j \cdot \log \theta_i^j$, where N_i^j represents the observed count for parameter θ_i^j in the training set. With these considerations we are now ready to present the main result of this subsection:

THEOREM 5.1. *The maximum likelihood estimator $\hat{\theta}_i^j$*

for a parameter θ_i^j in C_k that has type T_l is given by:

$$\hat{\theta}_i^j = \frac{N_i^j}{\sum_{\theta_{i'}^j \in T_l} N_{i'}^j} \cdot \frac{\sum_{\theta_{i'}^j \in T_l} N_{i'}^j}{\sum_{\theta_{i'}^j} N_{i'}^j}$$

Proof of Theorem 5.1 (Sketch): We introduce new variables $A = (A_l)_{1 \dots s}$ that represent the probability mass associated with type T_l in any of the distributions in C_k . As stated above, the log-likelihood function decomposes nicely in components corresponding to different C_k . With the newly introduced variables, our optimization problem can be restated as maximizing $f(C_k, A) = \sum N_i^j \cdot \log \theta_i^j$ subject to the constraints that $\sum_{\theta_i^j \in T_l} \theta_i^j = A_l$ for all types T_l and for any j^{th} distribution in C_k . In addition to these constraints, we also have $\sum_i \theta_i^j = 1$. Similarly to the previous theorems, it is easy to show that, if all counts are positive, then the function reaches a maximum inside the region defined by the constraints (if any count is zero, then the specific parameter doesn't even show up in the log-likelihood function and its estimator will be zero too). In this case, we also apply Lagrange Multipliers theory, introducing a lagrange multiplier for each constraint: λ_l^j for the first type of constraints (probability mass equalities) and λ^j for the second type (distributions should sum up to 1). Therefore, differentiating with respect to θ and A , the point that maximizes f inside the region given by the constraints should also verify:

$$(5.2) \quad N_i^j = \hat{\theta}_i^j \cdot (\lambda^j + \lambda_l^j) \quad \forall \theta_i^j \in T_l$$

$$(5.3) \quad \sum_j \lambda_l^j = 0 \quad \forall l$$

For a fixed j and l , summing up 5.2 for all i such that $\theta_i^j \in T_l$ we get:

$$(5.4) \quad \sum_{\theta_i^j \in T_l} N_i^j = A_l \cdot (\lambda^j + \lambda_l^j)$$

For a fixed l , summing up 5.4 for all j and using 5.3 we obtain:

$$(5.5) \quad \sum_{\theta_i^j \in T_l} N_i^j = A_l \cdot \sum_j \lambda^j$$

Further, summing 5.5 over all values of l and using the fact that the distributions sum up to 1, we can compute:

$$(5.6) \quad \sum_{i,j} N_i^j = \sum_j \lambda^j$$

Now we can use 5.6 in 5.5 to get A_l which is further used in 5.4 to obtain $\lambda^j + \lambda_l^j$ which, substituted in 5.2 will yield the formulas in the statement of the theorem. Our sketch of the proof is now complete. \square

5.2 Probability Ratio Sharing. In the previous subsection, we showed how to do learning when certain parameter types share their aggregate probability mass across different distributions. Now assume we want instead to enforce the constraint that the relative proportions of parameters in a certain type are the same for all different CPDs within a C_k . Next we describe a setting where this constraints may arise naturally. We would like again to model the word probabilities in two languages i.e. model $P(\text{word}|\text{language})$ for both languages and their corresponding sets of words. In this case, types can be: "words about computers" ("computer", "mouse", "monitor", "keyboard" in both languages) or "words about business", etc. In some countries computer use is more extensive than in others and one would expect the aggregate probability of "words about computers" to be different. However, it would be natural to assume that the relative proportions of the "words about computers" are the same within the different languages.

We keep the same notations in the previous subsection. There are two major differences from the setting presented in the previous subsection. First, in this case, we must have the same number of parameters of type T_l in each of the distributions in C_k . For example, "words about computers" can be "mouse", "keyboard" and "monitor" in both Spanish and English (if we have synonyms for a certain word, they can all be modelled as one cumulative parameter). This allows us to permute the parameters in the distributions in C_k such that corresponding parameters in T_l are located on the same position in each of the distributions. For example, we can assume $p(\text{mouse}|\text{English})$ and $p(\text{mouse}|\text{Spanish})$ are both on the first position in the two language models. This allows us to write that a specific position $i \in T_l$. Second, now there may be parameters that do not belong to any type T_l . For example, the expert may specify that only the "words about computers" preserve their relative probability ratios across the languages of interest. With these considerations, let us formalize the constraints that we are looking at:

Probability Ratio Sharing Assumption. For any fixed type T_l and for fixed $i_1, i_2 \in T_l$, the following holds:

$$\frac{\theta_{i_1}^j}{\theta_{i_2}^j} = \text{constant} \quad \forall j$$

The problem of finding the maximum likelihood

estimators subject to the above constraints can again be broken down in independent subproblems over the different subsets of CPDs C_k and the function to optimize for each C_k will be $f(C_k) = \sum N_i^j \cdot \log \theta_i^j$.

THEOREM 5.2. *The maximum likelihood estimator $\hat{\theta}_i^j$ for a parameter θ_i^j in C_k is given by:*

- a) if $i \in T_l$: $\hat{\theta}_i^j = \frac{\sum_{j'} N_i^{j'}}{\sum_{i' \in T_l, j'} N_{i'}^{j'}} \cdot \frac{\sum_{i' \in T_l} N_{i'}^j}{\sum_{i'} N_{i'}^j}$
- b) if θ_i^j does not have a type: $\hat{\theta}_i^j = \frac{N_i^j}{\sum_{i'} N_{i'}^j}$

Proof of Theorem 5.2 (Sketch): Again, we use Lagrange Multipliers theory to derive our estimators. Each CPD should sum up to 1 and that translates in the constraint $(\sum_i \theta_i^j) - 1 = 0$. Let the corresponding lagrange multiplier be λ^j . The *Probability Ratio Sharing Assumption* implies that there exist A_l^j and τ_i such that $\theta_i^j - A_l^j \cdot \tau_i = 0$ for all $i \in T_l$. A_l^j represent proportionality constants for distribution j for parameters of type T_l and τ_i are reference constants that, when multiplied with the proportionality constants yield the parameters on position i in each distribution. Let λ_i^j be the lagrange multipliers corresponding to the last type of constraints. Our new objective function becomes $f(C_k, A, \tau) = \sum N_i^j \cdot \log \theta_i^j$. When applying Lagrange Multipliers theory to our optimization problem, differentiating with respect to θ and the newly introduced A_l^j and τ_i , we obtain:

$$(5.7) \quad N_i^j = \hat{\theta}_i^j \cdot (\lambda^j + \lambda_i^j) \quad \forall i \in T_l$$

$$(5.8) \quad N_i^j = \hat{\theta}_i^j \cdot \lambda^j \quad \forall i \notin \cup T_l$$

$$(5.9) \quad \sum_j \lambda_i^j \cdot A_l^j = 0 \text{ or } \sum_j \lambda_i^j \cdot \hat{\theta}_i^j = 0 \quad \forall i \in T_l$$

$$(5.10) \quad \sum_j \lambda_i^j \cdot \tau_i = 0 \text{ or } \sum_{i \in T_l} \lambda_i^j \cdot \hat{\theta}_i^j = 0 \quad \forall j, l$$

For fixed j and l , summing up 5.7 for all $i \in T_l$, then using 5.10 we get:

$$(5.11) \quad \sum_{i \in T_l} N_i^j = \lambda^j \sum_{i \in T_l} \hat{\theta}_i^j$$

If we further sum over all l and use 5.8 we obtain:

$$(5.12) \quad \lambda^j = \sum_i N_i^j$$

Because $\frac{\hat{\theta}_i^{j_1}}{\hat{\theta}_i^{j_2}} = \frac{A_l^{j_1}}{A_l^{j_2}}$ for all j_1, j_2, l and $i \in T_l$, we can write:

$$(5.13) \quad \frac{A_l^{j_1}}{A_l^{j_2}} = \frac{\sum_{i \in T_l} \hat{\theta}_i^{j_1}}{\sum_{i \in T_l} \hat{\theta}_i^{j_2}} = \frac{\lambda^{j_2}}{\lambda^{j_1}} \cdot \frac{\sum_{i \in T_l} N_i^{j_1}}{\sum_{i \in T_l} N_i^{j_2}}$$

For a fixed $i \in T_l$ summing up 5.7 over all j and using 5.9 we have:

$$(5.14) \quad \sum_j N_i^j = \hat{\theta}_i^j \cdot (\lambda^j + \sum_{j' \neq j} \lambda^{j'} \cdot \frac{\hat{\theta}_i^{j'}}{\hat{\theta}_i^j})$$

Using 5.12 and 5.13 in 5.14 proves part a) of the theorem. Part b) follows from 5.8 and 5.12. The sketch of the proof is now complete. \square

6 Example

Previously published experiments involving learning with Module Networks, HMMs, DBNs or Context Specific Independence all support the theory presented in this paper since they are particular cases of our General Parameter Sharing Framework. However, the Parameter Sharing assumptions in the above models are at the level of either entire CPT or entire CPD. In this section we present experimental results showing the benefits of incorporating more fine-grained parameter sharing assumptions when training Bayes Multinets.

6.1 Email Modelling and Bayesian Multinets.

Automatic modelling of email documents is a problem of considerable interest, and has been studied as a means of automatically sorting email into a user's email subfolders, or into other topical categories such as "email spam" [11]. Given a set (population) of email, one natural way to define Bayesian Multinet subpopulations is to consider each distinct email *author* to be a generator of email from a different distribution, forming a different subpopulation. This is reasonable because different people use somewhat different vocabularies and figures of speech. At the same time, there are many content words that are shared (used in a similar proportion) by all authors when emails address specific topics such as meetings. The conditional probabilities of these words given that the email is about a meeting should therefore be specified as globally shared parameters across the subpopulations in the Bayesian Multinet.

To study this "meeting" email modelling problem, we generated synthetic data that captures the characteristics of a real email data set: the PW CALO email corpus, produced by people in a role-playing game at

SRI. Based on this corpus, we generated several artificial datasets. Each data point consists of a triple: *Author*, *Email* and *Topic* (Class). The Topic says whether or not the email is about a meeting. In all experiments we generated simulated email from six authors, using the same prior probabilities of an email belonging to an author as in the PW CALO corpus, and generating emails from each author to match that author’s topic priors in this corpus. For each given author and topic, we generate emails according to a ”Bag of Words” probability model, where each email contains between 0 and 150 words (to be consistent with the data observed in PW, even though we are not including stop words in our artificial dataset). The words are chosen from a vocabulary of 1070 words (same size as in PW). The word given topic probability models are generated randomly and differ from user to user, but also have some fraction of parameters in common (the so called shared parameters G_k in our framework). The fraction of shared parameters was varied in our experiments from 0 to 1. Creating this artificial data instead of using the real data allowed us to control and know which parameters are truly shared, allowed to assess the performance of our models in terms of KL Divergence from the true underlying distribution, and allowed us to control and study the effect of variations of different parameters in the true distribution (e.g. the fraction of parameters that are truly shared).

In our experiments we compare three models. First, a General Naive Bayes model (GNB) learned from all training examples. Second, a Bayesian Multinet (SSNB) in which each component network is a Naive Bayes model, and for which an oracle has indicated which parameters are shared when generating the data. Finally, a Bayesian Multinet (PSNB) identical to SSNB, but with no parameters shared among component networks. Note all three models are essentially Bayesian Multinets, conditioned on the email author which is observed in the header of the email. Each component network in each Bayes Multinet is a full naive Bayes model including both the Class/Topic variable and the word features in the email. One can think of GNB as a Bayesian Multinet where the component Bayes nets are copies of the GNB learned model. The only difference among the three is in the training procedure. They differ in their sharing of parameters (all shared in GNB, some shared in SSNB, not shared in PSNB). There is also a slight difference in the way we assign Dirichlet priors (we train all three models using MAP estimates, as is common when training Naive Bayes models from sparse data). In the case of GNB the effect is to increase each word count by one (equivalent to Dirichlet priors with all parameters equal to 2). In the case of PSNB and

SSNB, for each subpopulation the effect is to increase that subpopulation’s word counts by one. For PSNB, this is equivalent to Dirichlet priors with all parameters equal to 2 for each subpopulation, while for SSNB this is equivalent to assigning subpopulation-specific Dirichlet priors with parameters equal to 7 for shared parameters (7 is the number of subpopulations plus 1 in our experiments), and equal to 2 for local parameters.

6.2 Results and Discussion. We trained the three models while varying the number of training examples, and the fraction of word-given-class/topic model parameters that were shared (identical across authors). For each model, we measured the KL divergence $KL(T, M)$ of the learned model M to the true generating model T .

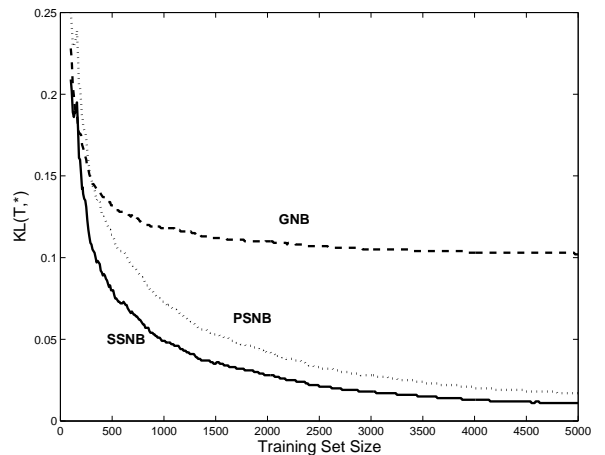


Figure 1: KL divergence of learned models with respect to correct model (T)

Figure 1 shows a plot of the KL divergence for each of the three models, as the number of training examples varies, keeping the fraction of general parameters constant at 0.5. As expected, $KL(T, *)$ decreases with increasing training set size for all three models (here $*$ stands for any possible model we are studying). However, SSNB outperforms the other two models across the entire range of training set size. It dominates PSNB especially at small training set sizes, because its shared global parameters allow it to produce lower-variance parameter estimates, especially when data is sparse. It dominates GNB especially with larger training sets, because it is capable of representing the correct model, whereas GNB considers a strictly smaller model class that does not contain the correct model. Note that asymptotically, as the training set size approaches infinity, the SSNB and PSNB models will both converge to the correct model, whereas GNB will not.

We also studied the impact of varying the fraction of global parameters in the true underlying probability model from 0 to 1, while holding the training set size constant at 1K. $KL(T,PSNB)$ is essentially constant as the fraction of true global parameters varies, because PSNB does not take advantage of parameter sharing. In contrast, both GNB and SSNB improve considerably with an increasing fraction of global parameters. Again, SSNB dominates the other two methods, as it can mix global and local parameters in its model.

7 Summary and Future Work

This paper presents a theoretical approach for incorporating several types of parameter related domain knowledge in learning procedures for graphical models parameterized by conditional probability tables. The main reason for taking advantage of such constraints is to alleviate learning from sparse data sets. First we describe in detail a General Parameter Sharing Framework that characterizes learning in a wide variety of graphical models like HMMs, DBNs or Module Networks. We develop sound procedures for learning in such models from both a Frequentist and a Bayesian point of view, from both complete and incomplete data, in the case when a domain expert can specify the structure of the graphical model and the parameters to be shared. Also, we prove some formal guarantees about our estimators and suggest ways for deciding among several potential Parameter Sharing Schemes. Second, we investigate a hierarchical extension of this framework based on Parameter Sharing Trees. Finally we present algorithms for using domain knowledge that specifies that certain groups of parameters share certain properties. In section 6, as an example, experimental results show that our shared parameter model achieved lower KL divergence to the true distribution when compared to two other classical methods: a global model and a standard Bayesian Multinet without parameter sharing.

This research suggests several directions for future work. First, we are developing proper conjugate prior distributions over the space of parameters that would allow us to develop MAP estimators in the setting presented in section 5. Second, we intend to explore the interaction between the different types of domain knowledge presented in this paper. Another interesting thing to investigate is how one can take advantage of parameter related domain knowledge to learn the structure of the graphical model.

Acknowledgements. This work was supported in part by DARPA research contract NBCD030010, and in part by a gift from Siemens Medical Solutions. We would like to thank the following people for

their useful comments during the development of this paper: William Cohen, Zoubin Ghahramani, Russ Greiner, John Lafferty, Andrew Moore and Sathyakama Sandilya.

References

- [1] C. Boutilier, N. Friedman, M. Goldszmidt, and D. Koller, *Context-specific independence in Bayesian networks*, Proceedings of 12th UAI (1996), pp. 115–123.
- [2] J. Cheng and R. Greiner, *Learning bayesian belief network classifiers: algorithms and system*, Proceedings of the Canadian Conference on Artificial Intelligence (2001).
- [3] D. M. Chickering, D. Heckerman, and C. Meek, *A Bayesian Approach to Learning Bayesian Networks with Local Structure*, Technical Report, MSR-TR-97-07, 1997.
- [4] N. Friedman and M. Goldszmidt, *Learning Bayesian Networks with Local Structure*, Proceedings of the 12th Annual Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers, CA, 1996, pp. 252–262.
- [5] N. Friedman, L. Getoor, D. Koller, and A. Pfeffer, *Learning Probabilistic Relational Models*, Proceedings of 16th IJCAI (1999), pp. 1300–1307.
- [6] D. Geiger and D. Heckerman, *Knowledge representation and inference in similarity networks and Bayesian multinets*, Artificial Intelligence, 82 (1996), pp. 45–74.
- [7] D. Geiger and D. Heckerman, *A characterization of the Dirichlet distribution through global and local parameter independence*, The Annals of Statistics, 25 (1997), pp. 1344–1369.
- [8] D. Golinelli, D. Madigan, and G. Consonni, *Relaxing the local independence assumption for quantitative learning in acyclic directed graphical models through hierarchical partition models*, Proceedings of Artificial Intelligence and Statistics (1999), pp. 203–208.
- [9] K. P. Murphy, *Dynamic Bayesian Networks: Representation, Inference and Learning*, PhD Thesis, UC Berkeley, Computer Science Division, 2002.
- [10] R. L. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech recognition*, Proceedings of the IEEE, 77 (2) (1989), pp. 257–286.
- [11] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, *A Bayesian Approach to Filtering Junk E-Mail*, AAAI Workshop on Learning for Text Categorization, Madison, WI, 1998.
- [12] E. Segal, D. Pe’er, A. Regev, D. Koller, and N. Friedman, *Learning Module Networks*, Proceedings of 19th UAI (2003), pp. 525–534.
- [13] C. Zhai and J. Lafferty, *A Study of Smoothing Methods for Language Models Applied to ad hoc Information Retrieval*, Proceedings of SIGIR (2001), pp. 334–342.