Probabilistic Graphical Models Parameter Learning with Transferred Prior and Constraints

Yun Zhou[†,‡], **Norman Fenton**[†], **Timothy M. Hospedales**[†], **Martin Neil**[†]
[†] Risk and Information Management (RIM) Research Group, Queen Mary University of London
[‡] Science and Technology on Information Systems Engineering Laboratory, National University of Defense Technology

Abstract

Learning accurate Bayesian networks (BNs) is a key challenge in real-world applications, especially when training data are hard to acquire. Two approaches have been used to address this challenge: 1) introducing expert judgements and 2) transferring knowledge from related domains. This is the first paper to present a generic framework that combines both approaches to improve BN parameter learning. This framework is built upon an extended multinomial parameter learning model, that itself is an auxiliary BN. It serves to integrate both knowledge transfer and expert constraints. Experimental results demonstrate improved accuracy of the new method on a variety of benchmark BNs, showing its potential to benefit many real-world problems.

1 INTRODUCTION

Directed probabilistic graphical models, also known as Bayesian networks (BNs), are a natural framework for describing probabilistic dependencies among variables in many real-world problems, such as medical symptom diagnosis (Velikova et al., 2014) and software defect prediction (Fenton and Neil, 2014). However, in problem domains with limited or no relevant training data, there are major challenges in accurately learning BN parameters (Friedman et al., 1999).

There are several methods for handling parameter learning with limited or no relevant data, described in a rich literature of books, articles and software packages, which are briefly summarized in (Druzdel and Van Der Gaag, 2000; Neapolitan, 2004; O'Hagan et al., 2006). Without considering any domain knowledge, the simplest learning approaches usually fail to accurately estimate parameters in a small dataset. To mitigate this problem, it may be possible to elicit numerical assessments from expert judgements, but this process is inefficient and error-prone.

Researchers have shown that experts tend to feel more comfortable providing qualitative or semi-numerical judgments (Feelders and van der Gaag, 2006) with less cognitive effort. Such judgments expressed as constraints between parameters of interest (e.g. "the probability of people getting cancer is smaller than 1%") are more reliable than numerical assessments, and have drawn considerable attention recently. In the work of (Zhou et al., 2014a), these kinds of constraints are modelled as nodes in an auxiliary BN model called MPL-C (Multinomial Parameter Learning model with Constraints), which includes nodes modelling training data statistics. The MPL-C improves parameter estimation accuracy by constraining the estimation with the expert constraints.

An alternative approach to improving BN learning in scarce data situations is to transfer knowledge from different but related BNs that may have more training data available (Luis et al., 2010). For example, transferring knowledge from the same medical diagnosis network learned in a different country. This can be effective if data for one or more sufficiently related source domains is available. However, the practical limitation is that transfer is contingent on availability of suitable related sources, and the relatedness of each source to the target task may not be known in advance. Estimating relatedness is thus important but challenging in practice, particularly when there are multiple potential sources of possibly varying relatedness.

While incorporating either parameter constraints or transfer learning from related data in source domains can improve parameter estimation accuracy, there exists no generic learning framework to synergistically exploit the benefits of both approaches. Achieving this is non-trivial because typical approaches to transfer (Luis et al., 2010) and to constrained learning (Zhou et al., 2014a) use very different formalisations. In this paper we generalise the state-of-the-art MPL-C model for learning with expert constraints to also exploit knowledge from related source domains via a bootstrap approach. The new model called MPL-TC (Multinomial Parameter Learning model with Transferred prior and Constraints) synergistically exploits both forms of external knowledge to improve learning performance in a target BN.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 provides a formalisation of the BN parameter learning problem. Section 4 introduces the MPL-C model and shows how it can be used to learn with parameter constraints. Section 5 describes our novel generalisation for constrained parameter learning with transfer from auxiliary sources (MPL-TC). Our method estimates relatedness to pick the best source to transfer from, and takes a bootstrap approach for generating target parameter priors from the source data. Section 6 gives empirical results on a set of benchmark datasets. Section 7 concludes the paper and discusses the future work.

2 RELATED WORK

Several models have been proposed to integrate parameter constraints and improve learning accuracy. The constrained convex optimization (CO) formulation (Altendorf et al., 2005; Niculescu et al., 2006; de Campos and Ji, 2008; de Campos et al., 2008; Liao and Ji, 2009; de Campos et al., 2009; Yang and Natarajan, 2013) is the most popular way to estimate the constrained parameters. In this setting, the algorithm seeks the globaly optimal estimation (maximal data log-likelihood) with respect to the parameter constraints. The parameters also can be estimated by Monte Carlo methods (Chang et al., 2008), where only the samples that satisfy the constraints are kept. Recently, auxiliary BN models (Zhou et al., 2014a,b) have been developed for solving this problem. This approach provides an extensible framework for parameter learning with additional information. However, these auxiliary models only use uniform parameter priors to regularize learning, which can be improved by informative priors (Neapolitan, 2004).

In the context of transfer learning in BNs, the multi-task framework of (Niculescu-mizil and Caruana, 2007) considers structure transfer. However, it assumes that all sources are equally related and simply learns the parameters for each task independently. The transfer framework of (Luis et al., 2010) (referred to as CPTAgg in this paper) measures the relatedness of tasks via calculating K-L divergence between target and source CPTs, and employs the heuristic weighted sum model for aggregating target and selected source parameters. The weights are proportional to the number of training samples. Finally, the study (Oyen and Lane, 2012) considers multi-task structure learning, again with independently learned parameters. Their model shows transfer performs poorly without knowledge of relatedness. However, they address this by using manually specified relatedness.

No previous work considers a generic BN parameter learning framework combining both transferred knowledge and constraints as discussed in this paper. Using the bootstrap approach for variability measurement in BNs is not new. For example, Friedman et al. (1999) study the robustness of network features based on DAGs learned on bootstrap resamples. Elidan (2011) uses bootstrap aggregation (bagging) to find a stable prediction model through improved computation of the log-likelihood score. However, to the best of our knowledge, this is the first work to use bootstrap to measure the variability of source MLEs and generate *TNormal*[1] parameter priors for transfer purpose.

3 BACKGROUND

A Bayesian network (BN) consists of three components: variables $V = \{X_1, X_2, X_3, ..., X_n\}$ corresponding to nodes of the BN, a set of numerical parameters θ of the variables in V, and a Directed Acyclic Graph (DAG) G encoding the statistical dependencies among the variables. For discrete variables, the probability distribution is described by a conditional probability table (CPT) that contains the probability of each value of the variable given each instantiation of its parents as defined by graph G. We write this as $p(X_i|\pi_i)$ where π_i denotes the set of parents of variable X_i in DAG G. Thus, the BN defines a simplified joint probability distribution over V given by:

$$p(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} p(X_i|\pi_i) \tag{1}$$

Let r_i denote the cardinality of the space of X_i, and $|\pi_i|$ represent the cardinality of the space of parent configurations of X_i. The k-th probability value of a conditional probability distribution $p(X_i|\pi_i = j)$ can be represented as $\theta_{ijk} = p(X_i = k|\pi_i = j)$, where $\theta_{ijk} \in \theta$, $1 \le i \le n$, $1 \le j \le |\pi_i|$ and $1 \le k \le r_i$.

In our BN parameter learning setting, we have data D combined with V and G to form the problem domain $\boldsymbol{\mathcal{D}} = \{V, G, D\}$. Within a domain $\boldsymbol{\mathcal{D}}$, the goal of parameter learning is to determine parameters for all $p(X_i|\pi_i)$. Given data D, the estimation of CPT parameters θ is conventionally solved by the Maximum Likelihood Estimation (MLE), $\hat{\theta} = \arg\max_\theta \log p(D|\theta)$. Let N_{ijk} be the number of data samples in D for which X_i takes its k-th value and its parents set π_i takes its j-th value, and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. The MLE estimate for each parameter is:

$$\hat{\theta}_{ijk} = \frac{N_{ijk}}{N_{ij}} \tag{2}$$

However, it is common (even for large datasets) that certain parent-child state combinations seldom appear, and MLE learning fails in this situation. Another classic parameter learning algorithm (Maximum a Posteriori, MAP) mitigates this by introducing a Dirichlet prior on θ so that: $\hat{\theta} = \arg\max_\theta \log p(D|\theta)p(\theta)$. This results in the MAP estimate $\hat{\theta}_{ijk} = \frac{N_{ijk}+\alpha_{ijk}}{N_{ij}+\alpha_{ij}}$. Intuitively, the hyperparameters α_{ijk} in the Dirichlet prior correspond to an expert's

[1]The abbreviation of *Truncated Normal* distribution.

guess of the corresponding virtual data counts. When there is no expert judgment, the K2 ($\alpha_{ijk} = 1$) or BDeu[2] ($\alpha_{ijk} = \dfrac{1}{|\pi_i| r_i}, \forall i, j, k$) priors are commonly used (Heckerman et al., 1995).

4 MPL-C MODEL

For parameter learning with constraints, the auxiliary multinomial parameter model has been proposed (MPL-C (Zhou et al., 2014a)). This method treats the parameters of interest and constraints as nodes in an auxiliary model. The parameter learning process is achieved via auxiliary model inference given observed data statistics and constraint conditions.

4.1 MODEL CONSTRUCTION

We use the estimation of probability distribution $p(X_i|\pi_i = j)$ (i.e., the *j*-th column[3] of the CPT associated with the variable X_i) as an example to illustrate the construction of the MPL-C model. Suppose there are r_i states, and the goal is to learn the r_i probability parameters $\theta_{ij1}, \ldots, \theta_{ijr_i}$ corresponding to these states. Assume we have N_{ijk} data observations of the *k*-th state ($1 \leq k \leq r_i$) and the total number of observations is N_{ij}. Then we can create a multinomial parameter learning BN model for each CPT column (shown schematically in Figure 1) to estimate parameters $\theta_{ij1}, \ldots, \theta_{ijr_i}$.

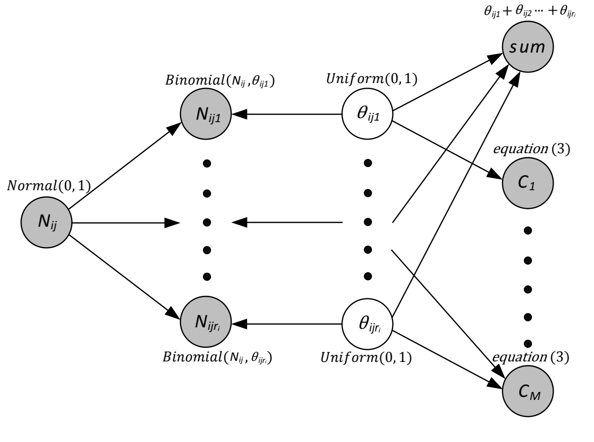

Figure 1: Graphical model representation of MPL-C with M constraints. For the constraint nodes, their equations follow the representations in equation 3. The gray nodes are observed during the inference.

Specifically, we start by creating an integer node named N_{ijk} (corresponding to N_{ijk} as defined above) for each k that is *Binomial* distributed. This node has two parents N_{ij} and θ_{ijk} to model the total number of trials and success probability in the *Binomial* distribution. The N_{ij} has a *Normal* distribution[4], which provides an infinite range for the total number of trials. The prior distribution of each θ_{ijk} is uniform between 0 and 1. Finally, there is an integer node *sum*, which is a shared child of θ_{ijk} ($k = 1, ..., r_i$). This node models the normalization constraint for the all success probabilities, i.e., that they should sum to 1.

4.2 INCORPORATING CONSTRAINTS

In real-world applications, many expert judgments can be described with linear inequality constraints and approximate equality constraints (Zhou et al., 2014a) having the following generic form:

$$\begin{cases} \beta_0 + \sum_{k=1}^{r_i} \beta_k \theta_{ijk} \leq 0 \\ |\theta_{ijk} - \theta_{ijk'}| \leq \varepsilon \; (0 < \varepsilon < 1) \end{cases} \tag{3}$$

Here the coefficients β_0, β_k ($1 \leq k \leq r_i$) are real numbers, and ε is an appropriate (small) positive value selected by the expert to represent $\theta_{ijk} \approx \theta_{ijk'}$.

Given that an expert has identified a number of constraints as defined above within a CPT column, then these constraints can be integrated as additional observed constraint nodes within the MPL model to generate a new model called MPL-C as shown in Figure 1.

Each constraint node C_m is a deterministic binary ($true/false$) node with expressions that specify the constraint relationships between its parents:

$$\begin{gathered} \textit{if}(\beta_0 + \textstyle\sum_{k=1}^{r_i} \beta_k \theta_{ijk} \leq 0, true, false) \\ \textit{if}(abs(\theta_{ijk} - \theta_{ijk'}) \leq \varepsilon, true, false) \end{gathered}$$

When the constraint is between a single parameter and a constant (i.e., $\beta_0 + \beta_k \theta_{ijk} \leq 0$), the constraint node will only have a single parent. Inference for the unobserved θ_{ijk} in this auxiliary model implements constrained MAP parameter learning for one CPT column of the target BN. In the next section we show how to generalise this framework to also take into account knowledge transfer from related source domains.

5 MPL-C MODEL WITH TRANSFERRED PRIOR

5.1 THE TRANSFER LEARNING MODEL

The idea behind transfer learning is to improve the accuracy of a target BN by making use of one or more related source BNs. For example, the target BN may be a model

[2]Bayesian Dirichlet likelihood equivalent uniform prior.

[3]Note in some other works, e.g., Netica BN software, each CPT row represents a discrete probability distribution given a parent configuration.

[4]This can be replaced with *Poisson* distribution to only allow positive integers. Because this root node is always observed with a valid number of trials during the inference, using *Normal* or *Poisson* distribution will produce the same results.

for diagnosis of a particular disease based on limited data in one district or country. If there are other (source) BNs with similar variables and objectives but from a different district or country, then it make sense to exploit such models to improve the accuracy of the target BN. Transfer learning does this by providing methods for both determining suitability of the data in the source and its transfer to the target.

The obvious practical limitation of transfer learning – which limits the applicability of all work in this area including this paper – is that the relatedness is never truly known. The necessary assumptions to overcome this introduce inevitable bias into the results. In this paper we assume there is at least one source domain that is sampled from similar distributions as the target, and that this can be transferred to help learn the target BN parameters. However determining relatedness in a data driven way means there is an inevitable confirmation bias in the sense that the source BNs most likely to be selected are those that most closely match the current target estimate. This limits the extent that the source can 'change' the target when it is more 'correct' than the current noisy target estimates.

If the chosen source and target are not sampled from similar distributions, directly applying parameters learned in another domain may be impossible or result in negative transfer: the underlying tasks may have major quantitative or qualitative differences (e.g., care procedures vary across hospitals). This limits the effectiveness of existing methods such as CPTAgg in (Luis et al., 2010). Our framework will address this by robustly measuring piecewise relatedness.

Given the scarce data for parameter estimation in the target domain $\boldsymbol{\mathcal{D}}^t = \{V^t, G^t, D^t\}$, the knowledge in available source domains should be mined to help the learning in the target domain. We aim to learn a target domain $\boldsymbol{\mathcal{D}}^t$ leveraging sources $\{\boldsymbol{\mathcal{D}}^s\}$ with potentially *piecewise* relatedness. Typically relatedness is computed at domain or instance level granularity, but for more flexible transfer we model relevance as varying within-domain. Thus transfer can still be exploited when different subsets of features/variables are relevant to different source domains. Thus we allow the heterogeneity $V^t \neq V^s$ and $G^t \neq G^s$, and transfer at the level of BN *fragments*.

***Definition* 1 BN fragment**. A Bayesian network of domain $\boldsymbol{\mathcal{D}}$ can be divided into a set of sub-networks (denoted *fragments*) $\boldsymbol{\mathcal{D}} = \{\boldsymbol{\mathcal{D}}_i\}$ by considering the graph G. Each fragment $\boldsymbol{\mathcal{D}}_i = \{V_i, G_i, D_i\}$ is a single root node or a node with its direct parents in the original BN, and encodes a single CPT from the original BN. The number of fragments is the number of variables in the original BN.

To achieve flexible BN parameter transfer, the target domain and source domains are all broken into fragments $\boldsymbol{\mathcal{D}}^t = \{\boldsymbol{\mathcal{D}}^t_i\}$, $\{\boldsymbol{\mathcal{D}}^s\} = \{\{\boldsymbol{\mathcal{D}}^s_{i'}\}\}$. Assuming for now no latent variables in the target domain, then each target fragment i can be learned independently $\hat{\theta}^t_i = \arg\max_{\theta^t_i} p(\theta^t_i | C^t_i, \boldsymbol{\mathcal{D}}^t_i, \{\{\boldsymbol{\mathcal{D}}^s_{i'}\}\})$. To leverage the bag of source domain fragments $\{\{\boldsymbol{\mathcal{D}}^s_{i'}\}\}$ in learning each θ^t_i, we consider each source fragment $\boldsymbol{\mathcal{D}}^s_{i'}$ as potentially relevant. Specifically, for each target fragment, every source fragment is evaluated for relatedness and the best fragment mapping is chosen. Once the best source fragment is chosen for each target, it will be used as parameter priors in the target MPL-C model. This auxiliary model can then be updated to infer the parameter posteriors given target data D^t_i, target constraints C^t_i and source networks $\{\boldsymbol{\mathcal{D}}^s\}$.

To realize this strategy, three issues must be addressed: 1) which source fragments are transferrable, 2) how to deal with variable name mapping, 3) how to quantify the relatedness of each transferrable source fragment in order to find the best one. We next address each of these issues in turn:

Fragment Compatibility For a target fragment i and putative source fragment i', we say they are *compatible* if they have the same structure and state space. That is, the same number of states and parents states. Additionally, if the target fragment contains parameter constraints C^t_i, the associated source parameter $\theta^s_{i'}$ should fall in the constrained value ranges $\Omega_{C^t_i}$, so

$$compatible(\boldsymbol{\mathcal{D}}^t_i, \boldsymbol{\mathcal{D}}^s_{i'}) = \begin{cases} 1 & if\ G^t_i = G^s_{i'}\ \&\ \theta^s_{i'} \in \Omega_{C^t_i} \\ & \quad \&\ dim(\theta^t_i) = dim(\theta^s_{i'}) \\ 0 & otherwise \end{cases}$$

where $dim(\theta^t_i) = dim(\theta^s_{i'})$ means $r^t_i = r^s_{i'}$ and $|\pi^t_i| = |\pi^s_{i'}|$. This assumption could be relaxed quite straightforwardly at the expense of additional computational cost. For example, if the target variable contains 2 states ($true$ and $false$), and one source variable contains 3 states ($high$, $medium$ and low), we can try multiple aggregations of the source states to generate three mappings to the target: 1) $true - high$ and $false - medium, low$; 2) $true - medium$ and $false - high, low$; 3) $true - low$ and $false - high, medium$.

Fragment Permutation Mapping For two fragments i and i' determined to be compatible, we still may not know the mapping between variable names. For example, if i has parents $[a, b]$ and i' has parents $[d, c]$, the correspondence could be $a - d, b - c$ or $b - d, a - c$. The function $permutations(G^t_i, G^s_{i'})$ returns an exhaustive list of possible mappings P_m that map states of i' to states of i.

Fitness Measurement To measure the relatedness between compatible target and source fragments $\boldsymbol{\mathcal{D}}^t_i$ and $\boldsymbol{\mathcal{D}}^s_{i'}$, we use Bayesian model comparison for two hypotheses: H_1 is the relevance hypothesis[5] that the source and target data share a common CPT, and H_0 (not H_1) is the indepen-

[5]Simplifying the fragment notation, so H_1 only refers the dependent hypothesis between $\boldsymbol{\mathcal{D}}^t_i$ and $\boldsymbol{\mathcal{D}}^s_{i'}$.

dent hypothesis that the source and target data have distinct CPTs. These two hypotheses are the outputs of our function $fitness(\boldsymbol{\mathcal{D}}^t_i, \boldsymbol{\mathcal{D}}^s_{i'}, p(H))$, and can be computed with:

$$\begin{aligned} p(H_1|D^s_{i'}, D^t_i) &\propto \int p(D^t_i|\theta_i)p(\theta_i|D^s_{i'}, H_1)p(H_1)d\theta_i, \\ p(H_0|D^s_{i'}, D^t_i) &\propto \int p(D^t_i|\theta^t_i)p(\theta^t_i|H_0)p(H_0)d\theta^t_i. \end{aligned} \tag{4}$$

For discrete data, the likelihood of H_1, integrating out the unknown CPTs θ_i, is the Dirichlet compound multinomial (DCM) or *Pólya* distribution:

$$p(D^t_i|D^s_{i'}, H_1) = \sum_{j=1}^{|\pi_i|} \left(\frac{\Gamma(\alpha^s_{i'j})}{\Gamma(N^t_{ij} + \alpha^s_{i'j})} \prod_{k=1}^{r_i} \frac{\Gamma(N^t_{ijk} + \alpha^s_{i'jk})}{\Gamma(\alpha^s_{i'jk})} \right) \tag{5}$$

where $\alpha^s_{i'jk}$ indicates the aggregate counts from the source domain and distribution prior, and $\alpha^s_{i'j} = \sum_k \alpha^s_{i'jk}$. Maximising $p(H_1|D^s_{i'}, D^t_i)$ over source networks s and fragment i' finds the fragment most likely to share the same generating distribution as the target, and thus the best source to transfer. Next we will discuss how to use the selected source data to help learn the target parameters.

5.2 THE MPL-TC MODEL

Given a target fragment and selected source fragment, the challenge is to fuse them in a robust manner. We solve this fusion problem in a Bayesian way – treating the transferred information as the target parameter priors. We refer to this framework as MPL-TC, which contains three main steps: 1) for each BN fragment in the target domain, find its closest source fragment and permutation in the source domain; 2) transfer the selected source fragment by converting the source data statistics into prior distributions of parameters in the target MPL-C model (see Section 4) and 3) perform the inference in this auxiliary model to learn the target parameters. The detail can be found in Algorithm 1.

Fragment Fusion To fuse the selected source fragment with the target fragment in the second step of our algorithm, we perform the bootstrap approach in the source to generate the priors of target parameters. Bootstrap is a resampling method to measure the quality of true samples (Duval, 1993). In this paper, we are interested in the quality of selected source parameters. We cannot access infinite training samples of selected source, instead we only have a sample of it (the selected best mapping source sample $D^s_{i'j}$), which means the MLE of $\theta^s_{i'jk}$ is not accurate.

From a specific subset of source samples, i.e., $D^s_{i'j}$, only one estimate of the MLE for a parameter of interest $\theta^s_{i'jk}$ can be obtained. In order to reason about the population, we need some sense of the variability of the estimated MLE. Thus, we apply the simplest bootstrap method – sampling from the $D^s_{i'j}$ to form a new sample (called a "resample" or bootstrap sample) that is also of size $|D^s_{i'j}|$. The bootstrap sample is taken from the original using sampling with replacement. This process is repeated multiple times (100 or 1000), and for each of these bootstrap samples we compute the MLE of $\theta^s_{i'jk}$ (each of these are called bootstrap estimates). We now have a set of bootstrap estimates, which are used to fit a *TNormal* distribution to encode how much the source MLE varies.

```
INPUT   : Target domain $\boldsymbol{\mathcal{D}}^t$, source domains $\{\boldsymbol{\mathcal{D}}^s\}$
          and target constraints $C^t$.
OUTPUT: The target parameters $\hat{\theta}^t$.
for each target fragment $i$ do
    for each source network $s$ and fragment $i'$ do
        if $compatible(\boldsymbol{\mathcal{D}}^t_i, \boldsymbol{\mathcal{D}}^s_{i'})$ then
            $P = permutations(G^t_i, G^s_{i'})$;
            for permutation $m = 1$ to $M$ do
                Measure relatedness:
                $fitness(\boldsymbol{\mathcal{D}}^t_i, P_m(\boldsymbol{\mathcal{D}}^s_{i'})) =$
                $p(H_1|D^t_i, P_m(D^s_{i'}))$
            end
        end
    end
    Find the best source fragment and permutation:
    $\arg\max_{i',s,m} p(H_1|D^t_i, P_m(D^s_{i'}))$
    for each parent state configuration $j$ do
        for each state value $k$ do
            $\left\{\theta^s_{i'jk}\right\} =$
            $bootstrap(100, @MLE, P_m(D^s_{i'j}))$
            Fit the $\left\{\theta^s_{i'jk}\right\}$ with
            $\zeta^s_{i'jk} = TNormal(\mu_{ijk}, \sigma_{ijk}, 0, 1)$
        end
        Generate the auxiliary model in the target:
        $\Psi^t_{ij} = mpltc(D^t_{ij}, C^t_{ij}, \zeta^s_{i'jk})$
        Inference to get the parameters estimation:
        $\hat{\theta}^t_{ij} = inference(\Psi^t_{ij})$
    end
end
return $\hat{\theta}^t = \left\{\hat{\theta}^t_{ij}\right\}$
```

Algorithm 1: Multinomial Parameter Learning with Transferred Prior and Constraints

In our MPL-TC approach, these *TNormal* distributions are used to replace the uniform parameter priors on θ^t_{ijk} (Figure 1) of MPL-C models in the target domain. Thus the transferred prior, target training samples and constraints are now all encoded in the target MPL-TC models (referred to as Ψ^t_{ij} in Algorithm 1). After observing the sources, the target data statistics $(N^t_{ij}, N^t_{ij1}, ..., N^t_{ijr_i})$ and available constraints (The constraint nodes are all observed with '$true$' values), we can update (by $inference(\cdot)$ function in Algorithm 1) these auxiliary models to get the target parameter posteriors:

$$p(\hat{\theta}^t_{ij1}, ..., \hat{\theta}^t_{ijr_t} | N^t_{ij}, N^t_{ij1}, ..., N^t_{ijr_i}, ... , C^t_1, ..., C^t_M, \zeta^s_{i'jk}, ..., \zeta^s_{i'jr_{i'}}, sum)$$

Because the auxiliary BNs are hybrid models, the update/inference is performed via a state-of-the-art dynamic discretization junction tree (DDJT) algorithm (Neil et al., 2007) that is implemented in AgenaRisk[6]. The time complexity of the inference is exponential in model treewidth, which restricts the applicability at some point. However, approximate inference could be used with dynamic discretization to improve the time efficiency.

5.3 ILLUSTRATIVE EXAMPLES

Bootstrap Fitting of TNormal Priors Figure 2 demonstrates an example of fitted *TNormal* distributions for MLE estimation $\theta^s_{i'jk} = 0.2$ learned from sample sizes 10 and 100. As we can see, although the parameter estimations of two source samples are the same, the estimation from the large sample are definitely more reliable than the estimation from the small sample, where the fitted *TNormal* distribution in $|D^s_{i'j}| = 100$ is much sharper than the distribution in $|D^s_{i'j}| = 10$. Moreover, the number of bootstrap replicates does not change the results much, and we use 100 replicates in all subsequent experiments.

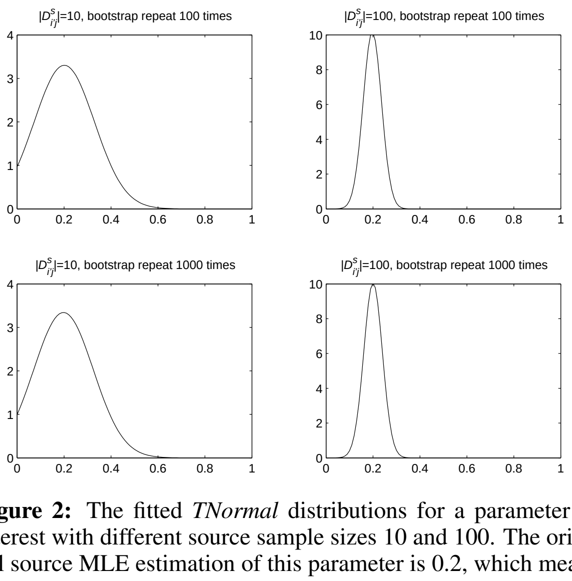

Figure 2: The fitted *TNormal* distributions for a parameter of interest with different source sample sizes 10 and 100. The original source MLE estimation of this parameter is 0.2, which means there are 2 and 20 appearances of this parameter in sample sizes 10 and 100 respectively. The bootstrap process in each case is repeated 100 and 1000 times.

Fragment Transfer Here we provide an illustrative example of our framework for fragment-based parameter transfer and the target parameter estimation: the target is a three node BN shown in the left part of Figure 3(a), and the source is an eight node BN shown in the right part of Figure 3(a). We aim to estimate the CPT of S^t, which has four parent state configurations – π^t_1, π^t_2, π^t_3 and π^t_4. As we can see, there are two source fragments ($\{T^s, L^s, E^s\}$ and $\{E^s, B^s, S^s\}$) which are *compatible* with target fragment (shown with dashed triangle in Figure 3(a)). Thus, there are four *permutations* of compatible source fragments (assuming binary parent nodes). All four of these options are then evaluated for *fitness*, and the best fragment and permutation is picked ($\{B^s, E^s, S^s\}$). In Figure 3(b), we generate four auxiliary BNs (MPL-TC model) for each target parameter column, the right part of Figure 3(b) shows the MPL-TC model of the first parameter column, which is used to estimate θ^t_{11} and θ^t_{12}. The constraints and data statistics in the target domain are modelled by nodes with gray color. The parameter priors (nodes with white color) are *TNormal* distributions fitted from source bootstrap samples. Finally, these priors are updated by observing the target data statistics and constraints to get the posterior parameter estimates.

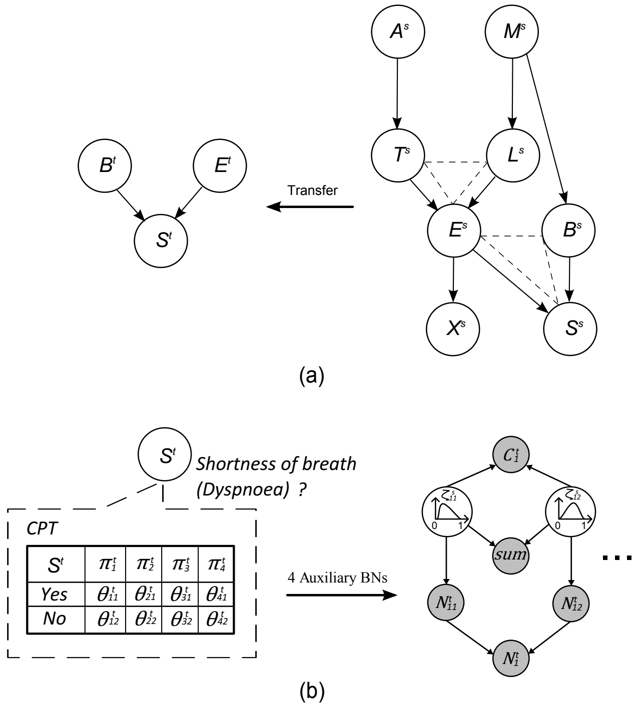

Figure 3: A simple example to show the framework of multinomial parameter learning with transferred prior and constraints. (a) The dashed triangle represents source fragments $\{T^s, L^s, E^s\}$ and $\{E^s, B^s, S^s\}$, which are *compatible* to the target fragment. (b) The structure representation of the MPL-TC model for estimating θ^t_{11} and θ^t_{12} is in the first target parameter column, whose parameter priors (θ^s_{11} and θ^s_{12}) are converted form the most fit source fragment $\{B^s, E^s, S^s\}$ via bootstrap.

6 EXPERIMENTS

6.1 EXPERIMENT SETTING

In all cases, we assume that the structure of the model is known and that the 'true' CPTs that we are trying to learn

[6]http://www.agenarisk.com/

are those that are provided as standard with benchmark BN models. For the purpose of the experiment we are not given these true CPTs but instead are given a limited number of sample observations which are randomly generated based on the true target CPTs. To introduce noise between the target and source for simulating varying relatedness, the source datasets are also sampled from the true CPTs but with 'soft' and 'hard' noise conditions: (1) soft: generate three source domains with 200, 300 and 400 sample sizes to simulate continuously varying relatedness among a set of sources; (2) hard: choose a portion (20%) of each source's fragments uniformly at random and randomise their data/CPTs to make them irrelevant. This results in a different subset of compatible but (un)related fragments in each source. Introducing these two types of sampling noise makes the sources similar but different to the target, and hence simulates the kind of source-target relations that may exist in practice.

The constraints are elicited from the true CPTs (so they are certainly correct) and randomly assigned to parameters in the network. Following the method of constraints generation in (Liao and Ji, 2009), for each true parameter θ_{ijk}, we create a constraint:

$$\min((1+\varepsilon)\theta_{ijk}, 1) \geq \theta^t_{ijk} \geq \max((1-\varepsilon)\theta_{ijk}, 0)$$

where the $\varepsilon = 0.05$ in the experiments. These elicited constraints are encoded in the MPL-TC auxiliary models, which are built with BN software AgenaRisk.

We compare our MPL-TC^{+5} against following algorithms and settings (the upper right superscript value associated with learning algorithms represents the number of constraints used in these algorithms):

- MLE and MAP, conventional BN parameter learning algorithms.
- MPL-C^{+5}, state-of-the-art parameter learning algorithm with five constraints (Zhou et al., 2014a).
- CPTAgg, state-of-the-art parameter transfer learning algorithm (Luis et al., 2010).
- MPL-TC^{+0}, our MPL-TC algorithm with zero constraints.

The resulting learned CPTs are evaluated against the true CPTs by using the K-L divergence measure (Kullback and Leibler, 1951). The smaller the K-L divergence is, the closer the estimated CPT is to the true CPT. Here the K-L divergence is locally measured for each CPT column and averaged over the whole model. This is to ensure that the fit of each distribution is equally weighted in the overall metric. Each experiment is repeated 10 times, and the results are reported with the mean and standard deviation of the K-L divergences between estimated and true CPTs.

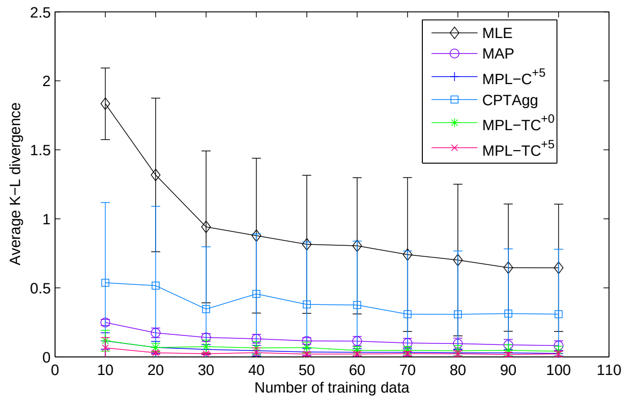

Figure 4: Parameter learning performance in the Cancer BN under different levels of data sparsity. Lower is better.

6.2 EXPERIMENTS ON CANCER BN

The Cancer BN (Korb and Nicholson, 2010) models the interaction between risk factors and symptoms for the purpose of diagnosing the most likely condition for a patient getting lung cancer. This BN contains 5 *Boolean* nodes, so each CPT column has just 2 parameters to learn; since the parameters sum to 1, each column has only one independent parameter. Hence there are 10 independent parameters to learn in the model. In the target domain, training samples under different sparsity levels (10 to 100 samples) are drawn from the ground-truth Cancer BN.

Overall Figure 4 presents the results of all learning algorithms under varying data volumes in the target Cancer BN. It is clear that the average K-L divergence of all learning algorithms decreases with increasing target sample size. With increasing sample sizes, the performance gap between the algorithms decreases[7]. Moreover, our MPL-TC^{+5} always outperforms all the other competitors, which demonstrates the effectiveness of our framework.

Considering models without parameter constraints (MLE, CPTAgg, MAP and MPL-TC^{+0}): MPL-TC^{+0} provides overall K-L divergence reductions (performance improvements) of 93.4%, 84.1% and 52.3% compared with MLE, CPTAgg and MAP respectively, thus demonstrating the efficacy of knowledge transfer.

After introducing 5 sampled constraints, MPL-TC^{+5} achieves even greater reductions in comparison with MLE, CPTAgg and MAP, which are 97.1%, 93.0% and 79.1% respectively. Due to the benefit of introducing parameter constraints, MPL-C^{+5} algorithm also outperforms MLE, CPTAgg and MAP. However, MPL-TC^{+5} still outperforms MPL-C^{+5} with 42.0% average K-L divergence reduction.

According to the results, the MPL-TC^{+5} greatly outperforms the conventional MLE and MAP algorithms, and the

[7]Given enough target training samples, the learning performance of all algorithms converge.

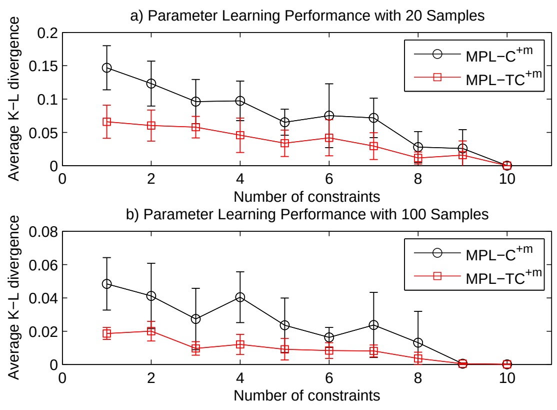

Figure 5: Performance of MPL-C and MPL-TC when varying the number of constraints ($m = 1, ..., 10$).

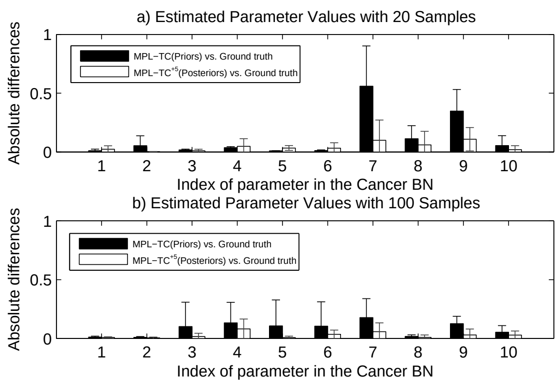

Figure 6: The differences between estimated probability values (MPL-TC(Priors) and MPL-TC^{+5}(Posteriors)) and ground truth for all parameters in the Cancer BN.

CPTAgg and MPL-C^{+5} that only use transfer or constraints alone. This demonstrates the complementarity of both constraints and sources of external knowledge when learning with scarce target data.

Varying Number of Constraints To investigate how the number of introduced constraints affect the learning performance of MPL-C and MPL-TC, we vary the number of sampled constraints in parameter learning (shown in Figure 5). As we can see, K-L divergence decreases with more constraints for both MPL-C and MPL-TC in both data sparsity settings. However, when the number of constraints is small, our MPL-TC greatly outperforms MPL-C due to the benefit of transferred parameter priors. When the number of constraints increases to 10 (every parameter is constrained), the learning results of MPL-C and MPL-TC both converge to zero in both settings.

Priors vs. Posteriors To provide insight into the mechanisms of our framework, we investigate the differences between MPL-TC(Priors) (transferred *TNormal* mean values) and MPL-TC^{+5}(Posteriors) (the updated parameter posteriors after inference given the target data and parameter constraints) for each parameter in the Cancer BN. The results are presented in Figure 6, where the heights of the bars represent the absolute differences between estimated values and true CPT values.

As we can see, the MPL-TC(Priors) shows inaccurate transfer in both two settings: parameters 7–9 in Figure 6(a) and parameters 3–7 and 9 in Figure 6(b). This is caused by the bias in target samples and noise in source domains. Therefore, the estimates of MPL-TC(Priors) are far from the true values, (average K-L divergence of 0.65 and 0.40 for sample sizes 20 and 100 respectively). However, after performing MAP learning in the MPL-TC^{+5} model, the MPL-TC^{+5}(Posteriors) reduces the average K-L divergence between the estimated values and true values to 0.09 and 0.03 respectively. These results demonstrate the robustness of the Bayesian learning in MPL-TC^{+5}, and the importance of systematically inferring the new parameters given available data and constraints. Next, we will compare the performance of all these algorithms in different BN parameter learning problems.

6.3 EXPERIMENTS ON STANDARD BNS

We evaluate the algorithms on 12 standard BNs[8] (details in Table 1). For each BN, 100 training samples and 5 constraints are drawn from the true CPTs in the target domain.

Overall Table 1 summarises the average K-L divergence per parameter. The best results are presented in bold. The statistically significant improvements of the best result over competitors are indicated with asterisks * (two-sample t-test at the default 5% significance level). In summary, the MPL-C^{+5} and MPL-TC^{+5} methods outperform conventional MLE and MAP in 11 out of 12 settings, the only exception is the learning performance in Weather BN, where the learning results of these methods converge with enough training samples[9]. These results demonstrate the benefit of learning with both sources of external knowledge. Compared with the state-of-the-art MPL-C^{+5}, MPL-TC^{+5} wins in every setting (including the Weather BN, where MPL-TC^{+5} achieves even smaller average K-L divergence – 0.018 of MPL-TC^{+5} vs. 0.020 of MPL-C^{+5}). Over all BNs, MPL-TC^{+5} gets 83.2%, 33.5% and 26.9% average reduction of K-L divergence compared with MLE, MAP and MPL-C^{+5} respectively.

Transfer vs. No Transfer Considering transfer learning only, both CPTAgg and MPL-TC^{+0} outperform conventional MLE, which demonstrate the benefit of introducing source domain knowledge. However, due to a simplistic relatedness model and CPT fusion heuristic, CPTAgg even fails to outperform MAP in some settings. In contrast, our MPL-TC^{+0} outperforms CPTAgg and MAP with 74.1%

[8]`http://www.bnlearn.com/bnrepository/`

[9]As shown in Table 1, the Weather BN only contains 9 parameters to learn, therefore 100 training samples are already enough to train a good model.

Table 1: Parameter learning performance (average K-L divergence) in 12 standard Bayesian networks.

Name	Nodes	Edges	Para	MLE	MAP	MPL-C^{+5}	CPTAgg	MPL-TC^{+0}	MPL-TC^{+5}
Alarm	37	46	509	2.36±0.10*	0.66±0.01*	0.61±0.02*	1.61±0.08*	**0.42** ±0.02	**0.42** ±0.01
Andes	223	338	1157	1.03±0.06*	0.17±0.01*	0.15±0.01*	0.65±0.05*	**0.08** ±0.00	**0.08** ±0.00
Asia	8	8	18	0.57±0.16*	0.34±0.04*	0.28±0.03*	0.31±0.05*	0.22±0.02*	**0.18** ±0.03
Cancer	5	4	10	0.86±0.35*	0.09±0.04*	0.07±0.05*	0.54±0.11*	0.05±0.01*	**0.03** ±0.01
Earthquake	5	4	10	1.50±0.82*	0.15±0.04*	0.13±0.03*	0.35±0.22*	0.11±0.01	**0.10** ±0.01
Hailfinder	56	66	2656	2.85±0.01*	0.46±0.00*	0.41±0.00*	1.98±0.01*	**0.31** ±0.01	**0.31** ±0.01
Hepar2	70	123	1453	3.18±0.13*	0.33±0.01*	0.33±0.01*	2.58±0.15*	0.30±0.01	**0.29** ±0.00
Insurance	27	52	984	1.95±0.18*	1.17±0.03*	1.07±0.03*	0.93±0.06*	**0.75** ±0.03	**0.75** ±0.02
Sachs	11	17	178	1.74±0.29*	0.78±0.04*	0.71±0.05*	0.98±0.08*	**0.50** ±0.03	**0.50** ±0.02
Survey	6	6	21	0.35±0.20*	0.05±0.01*	0.05±0.01*	0.24±0.15*	0.04±0.01	**0.03** ±0.01
Weather	4	4	9	**0.02**±0.02	0.03±0.00	**0.02**±0.00	**0.02**±0.00	**0.02**±0.00	**0.02**±0.00
Win95pts	76	112	574	3.59±0.07*	0.81±0.01*	0.78±0.02*	3.20±0.10*	0.67±0.02*	**0.64** ±0.01

and 31.2% average reduction of K-L divergence over all the settings. In addition, after introducing both transferred parameter priors and target constraints, our MPL-TC^{+5} shows additional improved learning performance over CPTAgg and MPL-TC^{+0} (75.0% and 3.5% average reduction of K-L divergence).

Importance of Transfer vs. Constraints As we can see, our MPL-TC^{+0} outperforms MPL-C^{+5}, which indicates the transferred prior is more helpful than a moderate number (i.e., 5) of constraints in improving parameter learning performance in these experiments. Given the burden of constraint elicitation in the real world, we used a realistic limited number of constraints. Of course if sufficient constraints were available, MPL-C would perform better (cf Figure 5) and this result would be reversed. But in this case, the transferred prior makes a greater contribution in improving performance – despite the noise process between source and target domain, and the imperfect estimation of relevance. This is especially in the larger BNs, where the constraints are scarcer relative to the number of parameters to learn. This also explains why MPL-TC^{+0} and MPL-TC^{+5} have similar results in the Alarm, Andes, Hailfinder, Insurance and Sachs BNs.

7 DISCUSSION AND CONCLUSIONS

When data is scarce, purely data driven BN parameter learning is inaccurate. The broad goal of this paper was to introduce a new method (MPL-TC) that is the first attempt at BN parameter learning incorporating both transfer learning and qualitative constraints in a complementary way. Using the public BN repository, we showed that learning performance was greatly improved in MPL-TC across a range of networks. In particular, we demonstrated that MPL-TC worked well in every data and constraint sparsity in the Cancer BN, and achieved the best performance in all BNs in the repository compared with other state-of-the-art algorithms.

We currently assume there is at least one relevant source. For each target fragment, we find the most relevant source fragment to generate target parameter priors. Transferring to a target fragment using information from >1 sources would be a straightforward modification of the current framework. However it would increase the risk of 'negative transfer' (Torrey and Shavlik, 2009) that could be detrimental to performance (if some apparently relevant sources used for transfer are actually false positives). This trade off between maximum exploitable transfer, and robustness to negative transfer is pervasive in transfer learning.

We discussed the limitations of all BN transfer learning approaches with respect to the fact that, in practice relatedness is hard to guarantee or estimate. Thus data-driven transfer (source selection) may be biased by inaccurate target data (resulting in bad choice of source and thus negative transfer) in extremely scarce settings. In the spirit of synergistically combining source data and constraints, available target constraints clearly provide an opportunity to guide and disambiguate transfer. In this paper, we only used target parameter constraints to exclude individual incompatible source fragments. Richer models for guiding transfer with constraints including cross-node constraints, source-domain constraints, and cross-domain constraints should be investigated in future.

8 ACKNOWLEDGMENTS

The authors would like to thank three anonymous reviewers for their valuable feedback. This work is supported by the European Research Council (ERC-2013-AdG339182-BAYES-KNOWLEDGE) and the European Union's Horizon 2020 research and innovation programme under grant agreement No 640891. YZ is supported by China Scholarship Council (CSC)/Queen Mary Joint PhD scholarships and National Natural Science Foundation of China (61273322, 71471174).

References

Altendorf, E.E., Restificar, A.C., Dietterich, T.G., 2005. Learning from sparse data by exploiting monotonicity constraints, in: Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence, pp. 18–26.

de Campos, C.P., Ji, Q., 2008. Improving Bayesian network parameter learning using constraints, in: Proceedings of the 19th International Conference on Pattern Recognition, pp. 1–4.

de Campos, C.P., Tong, Y., Ji, Q., 2008. Constrained maximum likelihood learning of Bayesian networks for facial action recognition, in: Proceedings of the 10th European Conference on Computer Vision. Springer, pp. 168–181.

de Campos, C.P., Zeng, Z., Ji, Q., 2009. Structure learning of Bayesian networks using constraints, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM. pp. 113–120.

Chang, R., Stetter, M., Brauer, W., 2008. Quantitative inference by qualitative semantic knowledge mining with Bayesian model averaging. Knowledge and Data Engineering, IEEE Transactions on 20, 1587–1600.

Druzdel, M., Van Der Gaag, L.C., 2000. Building probabilistic networks:"where do the numbers come from?". IEEE Transactions on knowledge and data engineering 12, 481–486.

Duval, R., 1993. Bootstrapping: A nonparametric approach to statistical inference. 94-95, Sage.

Elidan, G., 2011. Bagged structure learning of Bayesian network, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, pp. 251–259.

Feelders, A., van der Gaag, L., 2006. Learning Bayesian network parameters under order constraints. International Journal of Approximate Reasoning 42, 37–53.

Fenton, N.E., Neil, M., 2014. Decision support software for probabilistic risk assessment using Bayesian networks. IEEE software 31, 21–26.

Friedman, N., Goldszmidt, M., Wyner, A., 1999. Data analysis with Bayesian networks: A bootstrap approach, in: Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence, Morgan Kaufmann Publishers Inc.. pp. 196–205.

Heckerman, D., Geiger, D., Chickering, D.M., 1995. Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20, 197–243.

Korb, K.B., Nicholson, A.E., 2010. Bayesian Artificial Intelligence. CRC Press, New York.

Kullback, S., Leibler, R.A., 1951. On information and sufficiency. The Annals of Mathematical Statistics , 79–86.

Liao, W., Ji, Q., 2009. Learning Bayesian network parameters under incomplete data with domain knowledge. Pattern Recognition 42, 3046–3056.

Luis, R., Sucar, L.E., Morales, E.F., 2010. Inductive transfer for learning Bayesian networks. Machine learning 79, 227–255.

Neapolitan, R.E., 2004. Learning Bayesian networks. Pearson Prentice Hall.

Neil, M., Tailor, M., Marquez, D., 2007. Inference in hybrid Bayesian networks using dynamic discretization. Statistics and Computing 17, 219–233.

Niculescu, R.S., Mitchell, T., Rao, B., 2006. Bayesian network learning with parameter constraints. The Journal of Machine Learning Research 7, 1357–1383.

Niculescu-mizil, A., Caruana, R., 2007. Inductive transfer for Bayesian network structure learning, in: Proceedings of the 11th International Conference on Artificial Intelligence and Statistics, pp. 1–8.

O'Hagan, A., Buck, C.E., Daneshkhah, A., Eiser, J.R., Garthwaite, P.H., Jenkinson, D.J., Oakley, J.E., Rakow, T., 2006. Uncertain judgements: eliciting experts' probabilities. Wiley. com.

Oyen, D., Lane, T., 2012. Leveraging domain knowledge in multitask Bayesian network structure learning, in: Proceedings of the 26th AAAI Conference on Artificial Intelligence, pp. 1091–1097.

Torrey, L., Shavlik, J., 2009. Transfer learning. Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques 1, 242.

Velikova, M., van Scheltinga, J.T., Lucas, P.J., Spaanderman, M., 2014. Exploiting causal functional relationships in Bayesian network modelling for personalised healthcare. International Journal of Approximate Reasoning 55, 59–73.

Yang, S., Natarajan, S., 2013. Knowledge intensive learning: Combining qualitative constraints with causal independence for parameter learning in probabilistic models, in: Machine Learning and Knowledge Discovery in Databases. Springer, pp. 580–595.

Zhou, Y., Fenton, N., Neil, M., 2014a. Bayesian network approach to multinomial parameter learning using data and expert judgments. International Journal of Approximate Reasoning 55, 1252 – 1268.

Zhou, Y., Fenton, N., Neil, M., 2014b. An extended MPL-C model for Bayesian network parameter learning with exterior constraints, in: van der Gaag, L., Feelders, A. (Eds.), Probabilistic Graphical Models. Springer International Publishing. volume 8754 of *Lecture Notes in Computer Science*, pp. 581–596.