On Parameter Tying by Quantization

Li Chou, Somdeb Sarkhel
Department of Computer Science
The University of Texas at Dallas
{*lkc130030,sxs104721*}*@utdallas.edu*

Nicholas Ruozzi
Department of Computer Science
The University of Texas at Dallas
nicholas.ruozzi@utdallas.edu

Vibhav Gogate
Department of Computer Science
The University of Texas at Dallas
vgogate@hlt.utdallas.edu

Abstract

The maximum likelihood estimator (MLE) is generally asymptotically consistent but is susceptible to overfitting. To combat this problem, regularization methods which reduce the variance at the cost of (slightly) increasing the bias are often employed in practice. In this paper, we present an alternative variance reduction (regularization) technique that quantizes the MLE estimates as a post processing step, yielding a smoother model having several tied parameters. We provide and prove error bounds for our new technique and demonstrate experimentally that it often yields models having higher test-set log-likelihood than the ones learned using the MLE. We also propose a new importance sampling algorithm for fast approximate inference in models having several tied parameters. Our experiments show that our new inference algorithm is superior to existing approaches such as Gibbs sampling and MC-SAT on models having tied parameters, learned using our quantization-based approach.

Introduction

Weight (parameter) learning is a central problem in probabilistic graphical models. It is often expressed as the following maximum likelihood estimation (MLE) task: find an assignment of values to all parameters that maximizes the log-likelihood of the data. MLE is a popular weight learning formulation because it has several desirable theoretical properties including *consistency* (convergence in the limit to the correct value), and *asymptotic efficiency* (best-possible estimator in the limit). However, in practice, when the data size is small or the number of parameters is large (or both), MLE yields parameter estimates having high variance, and the resulting models have poor predictive (generalization) accuracy. To combat this, regularization methods such as ℓ_1 and ℓ_2 regularization, and their Bayesian counterparts Laplace and Gaussian priors, which reduce the variance by penalizing large parameter values, are often employed in practice.

In this paper, we consider an alternative regularization method: *parameter tying*, which forces several parameters of the graphical model to take the same value. Roughly speaking, since the sufficient statistics of the tied parameters can be combined, the number of samples used to estimate a specific parameter increases, which in turn reduces the variance (but increases the bias). Parameter tying is employed in a large variety of graphical models and their extensions such as hidden Markov models (Baum et al. 1970), dynamic Bayesian networks (Murphy 2002), conditional random fields (Lafferty, McCallum, and Pereira 2001), and statistical relational models (Getoor and Taskar 2007). However, a key feature of this existing work is that it assumes that the tied parameters are specified *apriori*. As a result, the full power of parameter tying as a regularization method has not yet been exploited. We deviate from the existing approach and seek methods that *automatically* tie the parameters by analyzing the data.

A straight-forward approach to solve the parameter tying problem is to express it as a constrained optimization problem. Given m parameters and a hyperparameter k, which bounds the maximum number of tied parameters and controls the amount of regularization, find a set of at most k equality constraints that are mutually exclusive (no two constraints have the same parameter) and exhaustive (all parameters are included) such that the likelihood is maximized. Unfortunately, this problem is computationally difficult as it includes structure learning as a special case.

To obviate this computational difficulty, we propose a greedy approach that ties parameters by *quantizing* the learned weights. Namely, we propose to find a many-to-one function f that maps m parameter values to a set having k values, such that the sum of the Euclidean distance between a parameter w and its quantized value $f(w)$ is minimized. This problem can be solved optimally using dynamic programming (Wang and Song 2011). We show that despite its simplicity, our new approach has several desirable theoretical properties and guarantees. In particular, we provide and prove novel *error bounds* which show that if the quantization is accurate then the difference between the average log-likelihoods of the original model and quantized model is quite small. Experimentally, we show that the quantized model often outperforms the untied model on a wide range of datasets.

A key benefit of parameter tying is that it introduces symmetries in the probabilistic model, and exploiting them makes inference easier and more efficient. To this end, we develop a novel importance sampling algorithm for fast, accurate sampling in parameter tied models. Our experiments conclusively show that our new sampling algorithm outperforms MC-SAT (Poon and Domingos 2006), a popular sampling algorithm that combines Markov Chain Monte Carlo sampling and satisfiability solution sampling, and Gibbs sampling (baseline) in the presence of tied parameters.

Background

We represent variables by capital letters (e.g., X), values in the domain of a variable by corresponding small letters (e.g., x) and an assignment of a value x to a variable X by $\bar{x}$. We represent sets of variables by bold capital letters (e.g., $\boldsymbol{X}$) and assignment of values to all variables in the set $\boldsymbol{X}$ by $\bar{\mathbf{x}}$. For simplicity of presentation, we will assume that all domains are binary unless otherwise noted.

A *log-linear probabilistic graphical model (PGM)*, denoted by $\mathcal{M}$ is a triple $\langle \boldsymbol{X}, \boldsymbol{F}, \boldsymbol{\theta} \rangle$ where $\boldsymbol{X} = \{X_1, ..., X_n\}$ is a set of variables, $\boldsymbol{F} = \{F_1, ..., F_m\}$ is a set of features and $\boldsymbol{\theta} = \{\theta_1, \dots, \theta_m\}$ is the set of weights (parameters) such that θ_i is the weight of feature F_i. $\mathcal{M}$ represents the following probability distribution

$$P_{\boldsymbol{\theta}}(\bar{\mathbf{x}}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\left(\sum_i \theta_i F_i(\bar{\mathbf{x}})\right)$$

where $F_i(\bar{\mathbf{x}})$ is 1 if the assignment $\bar{\mathbf{x}}$ evaluates F_i to true and 0 otherwise, and $Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \exp\left(\sum_i \theta_i F_i(\bar{\mathbf{x}})\right)$ is the normalization constant or the partition function.

Bayesian networks are directed graphical models in which each parameter θ_i has a probabilistic interpretation and equals the log of a conditional probability, and the partition function equals one. The two main inference problems over PGMs are: (1) marginal inference which is defined as computing the marginal probability of a variable given an assignment to a subset of variables (evidence) and (2) maximum-a-posteriori inference, which is defined as finding an assignment of values to all non-evidence variables that has the maximum probability. In this paper, we focus on the former.

Weight (Parameter) Learning

We assume that the PGM structure is known, namely the variables and the features are given while the weights are unknown and need to be estimated from data. We further assume that our dataset is such that all variables are observed and there are no missing values. Under these assumptions, given a training data set $\mathcal{D} = \{\bar{\mathbf{x}}^{(1)}, ..., \bar{\mathbf{x}}^{(D)}\}$ for variables $\boldsymbol{X}$, the weight learning task can be expressed as maximizing the following log-likelihood function.

$$\ell(\mathcal{D} : \boldsymbol{\theta}) = \log \prod_{i=1}^{D} P_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{(i)}) = \sum_{i=1}^{D} \log P_{\boldsymbol{\theta}}(\bar{\mathbf{x}}^{(i)})$$

This method of parameter learning or estimation is referred to as Maximum Likelihood Estimation (MLE). MLE has the desirable *consistency* property, which guarantees that the MLE solution converges to the true unknown parameter with high probability for sufficiently large samples. Under the assumptions given above, the log-likelihood function is concave. The optimum and thus the parameters (MLE solution) can be found using standard gradient descent methods. In Bayesian networks, the MLE solution can be computed in closed form (since each parameter is a conditional probability).

Importance Sampling

Importance Sampling (IS) (Liu 2001) is a Monte Carlo simulation technique, commonly used to evaluate the expectation, $\mathbb{E}_P[f(\boldsymbol{X})] = \sum_{\bar{\mathbf{x}} \in \boldsymbol{X}} f(\bar{\mathbf{x}}) P(\bar{\mathbf{x}})$ of a real function $f(\cdot)$ according to a probability distribution $P(\cdot)$. The general idea is to generate samples $\bar{\mathbf{x}}^{(1)}, ..., \bar{\mathbf{x}}^{(N)}$ from a proposal distribution Q, satisfying $f(\bar{\mathbf{x}}) > 0 \Rightarrow Q(\bar{\mathbf{x}}) > 0$, and then estimate $\mathbb{E}_P[f(\boldsymbol{X})]$ as follows

$$\mathbb{E}_P[f(\boldsymbol{X})] \approx \frac{\sum_{i=1}^{N} f(\bar{\mathbf{x}}^{(i)}) w(\bar{\mathbf{x}}^{(i)})}{\sum_{i=1}^{N} w(\bar{\mathbf{x}}^{(i)})}$$

where $w(\bar{\mathbf{x}}^{(i)}) = \frac{P(\bar{\mathbf{x}}^{(i)})}{Q(\bar{\mathbf{x}}^{(i)})}$ is called the importance weight of the sample $\bar{\mathbf{x}}^{(i)}$. Note that the importance weight needs to be known only up to a multiplicative constant. Importance sampling can be used to estimate both the partition function (normalizing constant) as well as one variable marginals. In this paper we focus on computing the single variable marginal probability, which can be estimated as

$$\widehat{P}_N(\bar{x}) = \frac{\sum_{i=1}^{N} \mathbf{1}\{\bar{\mathbf{x}}^{(i)}\} w(\bar{\mathbf{x}}^{(i)})}{\sum_{i=1}^{N} w(\bar{\mathbf{x}}^{(i)})} \tag{1}$$

where $\mathbf{1}\{\bar{\mathbf{x}}\} = 1$ iff $\bar{\mathbf{x}}$ contains the assignment $\bar{x}$, and 0 otherwise, and $w(\bar{\mathbf{x}}) = \frac{\exp(\sum_i \theta_i F_i(\bar{\mathbf{x}}))}{Q(\bar{\mathbf{x}})}$. In this paper, we will make the standard assumption that Q is a Bayesian network (Fung and Chang 1989; Ortiz and Kaelbling 2000; Cheng and Druzdzel 2001; Gogate 2009), since it is easy to generate independent samples from a Bayesian network using logic (forward) sampling (Pearl 1988). The quality of estimation, namely the accuracy of $\widehat{P}_N(\bar{x})$ is highly dependent on how far Q is from P, and as a result most of the research on importance sampling is about designing a good Q. In this paper, we design a Q for parameter tied PGMs.

Quantization, Clustering and Partition

Quantization is the process of mapping a larger set of real numbers to a smaller set. Formally, given two sets of real numbers $\boldsymbol{w}$ and $\boldsymbol{v}$, a quantization function $\mathcal{Q}$, called a *quantizer*, is a surjective function from elements of $\boldsymbol{x}$ to elements of $\boldsymbol{y}$ where $|\boldsymbol{x}| \geq |\boldsymbol{y}|$, such that $|x_i - \mathcal{Q}(x_i)| \leq \epsilon_i$ for all $x_i \in \boldsymbol{x}$. Here ϵ_i is referred to as error of quantization. Often we are interested in $\mathcal{Q}$ which minimizes the average error of quantization (i.e., $\frac{1}{|\boldsymbol{x}|} \sum_i |x_i - \mathcal{Q}(x_i)|$).

The optimal quantization problem is closely related to the optimal k-means clustering problem which is defined as follows: given a set of observations $\boldsymbol{x} = \{x_1, ..., x_n\}$, find a k-partition $\mathcal{S} = \{\boldsymbol{S_1}, ..., \boldsymbol{S_k}\}$ of x (namely, $\forall i$ $\boldsymbol{S_i} \subseteq \boldsymbol{x}$ and $\cup_{i=1}^{k} \boldsymbol{S_i} = \boldsymbol{x}$) such that following objective function is minimized

$$\arg\min_{\mathcal{S}} \sum_{i=1}^{k} \sum_{x \in \boldsymbol{S_i}} ||x - \mu_{\boldsymbol{S_i}}||_{\mathfrak{d}}$$

where $||\cdot||_{\mathfrak{d}}$ is a distance measure and $\mu_{\boldsymbol{S_i}}$ (called the *cluster center*), is the mean of elements in $\boldsymbol{S_i}$. The most popular distance measure is *Euclidean distance*.

A function $\mathcal{Q} : \boldsymbol{x} \rightarrow \boldsymbol{\mu}$, defined as $\mathcal{Q}(x) = \mu_{\boldsymbol{S_i}}$ iff x is in cluster $\boldsymbol{S_i}$, where $\boldsymbol{\mu} = \{\mu_{\boldsymbol{S_1}}, ..., \mu_{\boldsymbol{S_k}}\}$, is clearly a quantizer of $\boldsymbol{x}$. Hence, we can use any clustering algorithm (e.g., the k-means algorithm) for quantization. In this work, it turns out that we will only need to solve one dimensional clustering problems, which can be solved in polynomial time, $O(m^2k)$, using dynamic programming (Wang and Song 2011). In this paper, we leverage this algorithm for finding optimal quantizations.

Quantization for Weight Learning

We define a parameter tied graphical model $\mathcal{M}_t$, as a tuple $\langle \boldsymbol{X}, \boldsymbol{\theta}, \mathcal{C} \rangle$, where $\boldsymbol{X}$ is the set of variables, $\boldsymbol{\theta}$ is the parameter set, and $\mathcal{C}$ is a set of equality constraints of the form $\theta_i = \theta_j$, for some $\theta_i, \theta_j \in \boldsymbol{\theta}$. Since equality constraints are equivalence relations, $\mathcal{C}$ induces a partition over $\boldsymbol{\theta}$ where each part of the partition corresponds to an equivalence class. We are interested in learning the optimal parameter tied model $\mathcal{M}_t^*$ from data. This problem can be defined as follows. Given training data $\mathcal{D}$ on variables $\boldsymbol{X}$, find the constraint set $\mathcal{C}$ and parameters $\boldsymbol{\theta}$ such that $\mathcal{C}$ induces a k-partition on $\boldsymbol{\theta}$ and the (log)likelihood of data is maximized.

Solving the aforementioned optimization problem is hard because it requires searching over all possible constraint sets, which is clearly impractical. In particular, the number of possible constraint sets of size k for m parameters equals the number of partitions of size k for a set of size m (denoted by $\left\{ {m \atop k} \right\}$). This number is given by the so-called *Stirling numbers of the second kind* which grows exponentially with m. The total number of partitions of a set is given by the *Bell number*, $B_m = \sum_{k=1}^{m} \left\{ {m \atop k} \right\}$.[1]

Our approach. To remedy this computational difficulty, we propose the following greedy approach. First, we learn the parameters of the log-linear model using MLE. Then, we quantize these parameters to k levels using the one-dimensional k-means clustering algorithm.

Error bound for Quantization

Although, our approach is simple and straightforward, we show next that it will yield models that have high log-likelihood score yet fewer parameters under the assumption that the quantization error is small. Formally, let $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_m)$ denote the parameters of a log-linear model learned from a dataset $\mathcal{D}$ having D examples. Without loss of generality, we assume that all weights are positive. Let $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_k)$, $k \leq m$ be a quantization of $\boldsymbol{\theta}$ with respect to the quantizer $\mathcal{Q}$ between $\boldsymbol{\theta}$ and $\boldsymbol{\mu}$ such that for all $\theta_i \in \boldsymbol{\theta}$, the following holds

$$|\theta_i - \mathcal{Q}(\theta_i)| \leq \epsilon$$

[1]The fact that the optimization problem is computationally difficult also follows from the observation that it includes structure learning as a special case where the number of true features is $k-1$.

where $\epsilon \geq 0$ is a small constant. Let $\ell(\boldsymbol{\theta} : \mathcal{D})$ and $\ell(\boldsymbol{\mu} : \mathcal{D})$ denote the log-likelihoods of $\mathcal{D}$ with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\mu}$ respectively. Then, we can prove that the difference between the average log-likelihood scores of the quantized model and the original model is bounded by $2m\epsilon$. Formally, (*proof is available in the extended version*)

Theorem 1.

$$\frac{1}{D}\left(\ell(\boldsymbol{\theta} : \mathcal{D}) - \ell(\boldsymbol{\mu} : \mathcal{D})\right) \leq 2m\epsilon.$$

As the number of quantization levels (k) increases, the quantization error ϵ decreases, and vanishes when $k = m$. As a result, the bound specified in Theorem 1 becomes tighter, and our greedy learning approach yields more accurate results while using a smaller number of parameters.

Relearning

The log-likelihood score of the quantized log-linear model can be further improved by simply relearning the model, treating all parameters that are quantized to the same value as tied. Formally, since the quantizer $\mathcal{Q}$ induces a k-partition on the original parameter set $\boldsymbol{\theta}$, we set up the learning problem as follows

$$\begin{aligned} &\underset{\boldsymbol{\theta}}{\text{maximize}} \;\; \ell(\boldsymbol{\theta} : \mathcal{D}) \\ &\text{subject to} \;\; \{\theta_i = \theta_j | \theta_i, \theta_j \in \boldsymbol{\theta}, \mathcal{Q}(\theta_i) = \mathcal{Q}(\theta_j)\}. \end{aligned} \tag{2}$$

The optimization problem given above has a concave objective function, and therefore has a single maximum, and therefore can be solved using the same approaches (e.g., gradient descent) that are used to solve the MLE problem. As mentioned earlier, the key benefit of the formulation in Eq. (2) is that it has fewer (unique) parameters and therefore the data statistics are estimated using a larger sample size than the ones used in the unconstrained (MLE) version of this problem. From standard sampling theory (Liu 2001), this reduces the variance of the estimates.

Proposition 1. *Let $\boldsymbol{\mu}$ denote set of quantized parameters, $\mathcal{Q}$ be the corresponding quantizer and $\boldsymbol{\rho}$ be the relearned parameters (optimal solution of the optimization problem given in Eq.* (2)*), then*

$$\ell(\boldsymbol{\rho} : \mathcal{D}) \geq \ell(\boldsymbol{\mu} : \mathcal{D}).$$

Proof. Since $\boldsymbol{\mu}$ is a feasible solution of the problem given in Eq. (2) (it satisfies all the the constraints) and $\boldsymbol{\rho}$ is the optimal solution, it follows that $\ell(\boldsymbol{\rho} : \mathcal{D}) \geq \ell(\boldsymbol{\mu} : \mathcal{D})$. $\square$

Slice Importance Sampling

We observe that parameter tying introduces symmetries in the network which can be exploited to design a fast, approximate inference algorithm. In this section, we propose one such algorithm, which is based on slice importance sampling (Neal 2000; Gogate and Domingos 2010).

In slice importance sampling, the proposal distribution is defined over the features rather than over the variables. For instance, we can define a proposal distribution

$$Q(\boldsymbol{F}) = Q(F_1) \prod_{i=2}^{m} Q(F_i|F_1, \ldots, F_{i-1})$$

over the set of features $\boldsymbol{F}$. Sampling each feature in order from Q yields a $0/1$ assignment to the features where 0 indicates that the negation of the feature is true while 1 indicates that the feature is true. We can consider this $0/1$ assignment as a slice over the possible assignments to the variables in the following sense. All (variable) assignments that satisfy all features assigned to 1 and all negations of features assigned to 0 will have the same probability. Uniformly sampling over this subset of variable assignments (the slice) gives us the required sample. The benefit of slice sampling is that all variable assignments having the same probability in the distribution represented by the log-linear model have the same probability (assuming uniform sampling over the slice can be done) in the proposal distribution. This reduces the variance (Neal 2000; Gogate and Domingos 2010).

In our new algorithm, our main idea is to define the proposal distribution over the tied features rather than over the individual features, further reducing the variance. We first create a set of 'super-features' $\boldsymbol{\mathcal{G}}$, such that each of them contains features with tied parameters, i.e.,

$$\boldsymbol{G_i} = \{F_j | F_j \text{ has parameter } \theta_i\}, \tag{3}$$

and defined a proposal over them as follows (using the chain rule):

$$Q(\boldsymbol{\mathcal{G}}) = Q(\boldsymbol{G_1}) \prod_{i=2}^{k} Q(\boldsymbol{G_i}|\boldsymbol{G_1}, \ldots, \boldsymbol{G_{i-1}}).$$

However, one problem with defining a proposal over the super-features is that the number of values each super-feature $\boldsymbol{G_i}$ can take is exponential in the number of features it contains, namely exponential in $|\boldsymbol{G_i}|$. To compactly represent these exponentially many assignments, we use the so-called *counting assignments* (Milch et al. 2008; Jha et al. 2010) as follows. We partition the assignments to the features in the set $\boldsymbol{G_i}$ into $|\boldsymbol{G_i}|+1$ subsets where the j-th subset contains all assignments in which exactly j features in $\boldsymbol{G_i}$ are assigned to true and the remaining are assigned to false. Thus, each super-feature $\boldsymbol{G_i}$ can take $|\boldsymbol{G_i}| + 1$ values, yielding exponential reduction in complexity.

Another benefit of using counting assignments is that they yield better proposal distributions. In particular, the set of valid counting assignments to all super-features partitions the set of assignments to the variables into equiprobable subsets, where each subset is composed of assignments to variables that satisfy a counting assignment to all super-features. In an ideal proposal distribution all such assignments must have the same probability, and defining the proposal over the counting assignments preserves this property.

Example 1. *Consider a log-linear model having three features $F_1 = a \vee b$, $F_2 = b \vee c$ and $F_3 = a \vee c$ and three binary variables a, b, and c. Let the features F_1 and F_2 have the same weight θ_1 and θ_2 be the weight associated with F_3. We define two super-features: $\boldsymbol{G_1} = \{F_1, F_2\}$ and $\boldsymbol{G_2} = \{F_3\}$. $\boldsymbol{G_1}$ has four possible assignments: $\{(F_1 = 0, F_2 = 0), (F_1 = 0, F_2 = 1), (F_1 = 1, F_2 = 0), (F_1 = 1, F_2 = 1)\}$ but only three possible counting assignments $\{0, 1, 2\}$, where 0, 1 and 2 correspond to the subset of assignments $\{(F_1 = 0, F_2 = 0)\}$, $\{(F_1 = 0, F_2 = 1), (F_1 = 1, F_2 = 0)\}$ and $\{(F_1 = 1, F_2 = 1)\}$ respectively. The reader can verify that the assignments to the the variables that satisfy either $(F_1 = 0, F_2 = 1)$ or $(F_1 = 1, F_2 = 0)$ have the same probability.*

Algorithm 1 Tied Weight Importance Sampling

```
Input: A log-linear model $\mathcal{M} = \langle \boldsymbol{X}, \boldsymbol{F}, \boldsymbol{\mu} \rangle$ with $k$ unique
    weights, Number of samples $N$
Output: Importance weighted samples
 1: Create one super-feature $\boldsymbol{G_i}$ for each parameter $\mu_i$
 2: Construct a proposal distribution $Q(\boldsymbol{G})$ over the super-
    features
 3: for $s$ = 1 to $N$ do
 4:     $S = \emptyset$; $w^{(s)} = 1$
 5:     for $i = 1$ to $k$ do
 6:         $j_i \sim Q(\boldsymbol{G_i}|\boldsymbol{G_1}, \ldots, \boldsymbol{G_{i-1}})$
 7:         Add $j_i$ randomly selected features from $\boldsymbol{G_i}$ to $S$
 8:         Add the negation of the features from $\boldsymbol{G_i}$ not
    selected in the previous step to $S$
 9:         $w^{(s)} = w^{(s)} \times \binom{|\boldsymbol{G_i}|}{j_i} \frac{\exp(j_i \mu_i)}{Q(\boldsymbol{G_i}|\boldsymbol{G_1},\ldots,\boldsymbol{G_{i-1}})}$
10:     end for
11:     Sample $\bar{\boldsymbol{x}}^{(s)} \sim \mathcal{U}_{SAT}(S)$
12:     $w^{(s)} = w^{(s)} \times \#S$
13: end for
14: return $(\bar{\boldsymbol{x}}^{(s)}, w^{(s)})$ for $s = 1$ to $N$
```

However, counting assignments introduce the following problem. To generate a sample, we need to generate an assignment to variables uniformly at random from the subset of variable assignments that satisfy the counting assignment. Unfortunately, this problem is extremely challenging and to our knowledge no general-purpose algorithms exist for it. To alleviate this computational difficulty, we propose to use the following proposal distribution:

$$Q(F_1, \ldots, F_{|\boldsymbol{G_i}|} | \forall j \; F_j \in \boldsymbol{G_i}, \boldsymbol{G_i} = t) = \frac{1}{\binom{|\boldsymbol{G_i}|}{t}} \tag{4}$$

Sampling from this proposal distribution yields a $0/1$ assignment to the features and the only problem that remains to be solved is generating an assignment to variables uniformly at random from the subset of variable assignments that satisfy the given assignment to the features. This problem can be reduced to the uniform solution sampling problem, a well-researched problem for which a number of general-purpose solvers and techniques exist (Gogate and Dechter 2011; Wei, Erenrich, and Selman 2004; Gogate 2009).

Algorithm 1 formally describes our proposed slice importance sampling algorithm. The sampling process begins by constructing a proposal distribution Q over the super-features $\boldsymbol{G_i}$ associated with the same parameter μ_i. Then we sample assignments (j_i) for each super-feature $\boldsymbol{G_i}$ from the proposal Q in a selected order (step 6). Then we select j_i random features from $\boldsymbol{G_i}$ and set their assignment as 1. The rest of the features in $\boldsymbol{G_i}$ are assigned 0. This sampled $0/1$ assignment to all features defines a satisfiability problem S. Solutions of this satisfiability problem correspond to the subset of variable assignments that have the same probability. Thus to generate a sample, all we have to do is uniformly sample the solutions of S (step 11). For this (procedure $\mathcal{U}_{SAT}$), we can use uniform solution samplers such as SampleSAT (Wei, Erenrich, and Selman 2004) and SampleSearch (Gogate and Dechter 2011). In our experiments, we used the latter. The weight $w^{(s)}$ of the generated sample is proportional to the ratio between the probability of generating the sample from $\mathcal{M}$ and the probability of generating it from the proposal distribution and is computed iteratively in Steps 9 and 12.

In our experiments, we have used a simple proposal which factorizes independently over the super-features, i.e.,

$$Q(\boldsymbol{\mathcal{G}}) = \prod_{i=1}^{k} Q(\boldsymbol{G_i}).$$

Let $\mu_1, \ldots, \mu_k$ be the parameters and let $\boldsymbol{G_i}$ denote the super-feature that is associated with μ_i. We define $Q(\boldsymbol{G_i})$ as the following binomial distribution

$$Q(\boldsymbol{G_i} = t) \triangleq \binom{|\boldsymbol{G_i}|}{t} \frac{e^{t\mu_i}}{(1 + e^{\mu_i})^{|\boldsymbol{G_i}|}} \tag{5}$$

where $t \in \{0, \ldots, |\boldsymbol{G_i}|\}$. Notice that the binomial is defined over $|\boldsymbol{G_i}| + 1$ points as opposed to the conventional proposal which would have been defined over $2^{|\boldsymbol{G_i}|}$ points.

Experiments

We evaluated the performance of our quantized approach on both learning and inference tasks using several publicly available benchmark datasets from the UAI 2008 probabilistic inference competition repository (http://graphmod.ics.uci.edu/uai08). All experiments were performed on quad-core Intel i7 based machines with 16GB of RAM running Ubuntu.

Weight Learning

First, we compared our quantized tied weight learning algorithms to the MLE with a Laplacian prior on a collection of Bayesian network learning problems. For each selected Bayesian network, we used forward sampling to generate 100 sets of 6,000 training, 2,000 validation and 2,000 test data points. Using the training data, we learned three models corresponding to the MLE, the quantized MLE, and a MLE obtained by relearning after quantization for different values of k. Performance of each learning technique was evaluated using the average log-likelihood over the test set. We consider three kinds of Bayesian networks: two-layered noisy-or Bayesian networks (Savicky and Vomlel 2009), relational Bayesian networks constructed from the Primula tool (UCLA), and grid networks (Sang, Beame, and Kautz 2005). The various networks and the respective number of parameters in the networks are shown in Table 1.

Network	Total # Param.	True k	Est. k	Max Error MLE	Max Error Q	Max Error RL
bn2o	20510	20491	500	0.339	0.339	0.299
students	1308	13	20	0.141	0.141	0.140
grid	1089	596	400	0.0186	0.0186	0.0181
friends	3899	6	10	0.008	0.008	0.007

Table 1: Network information and error analysis (MLE, Quantized and Relearned

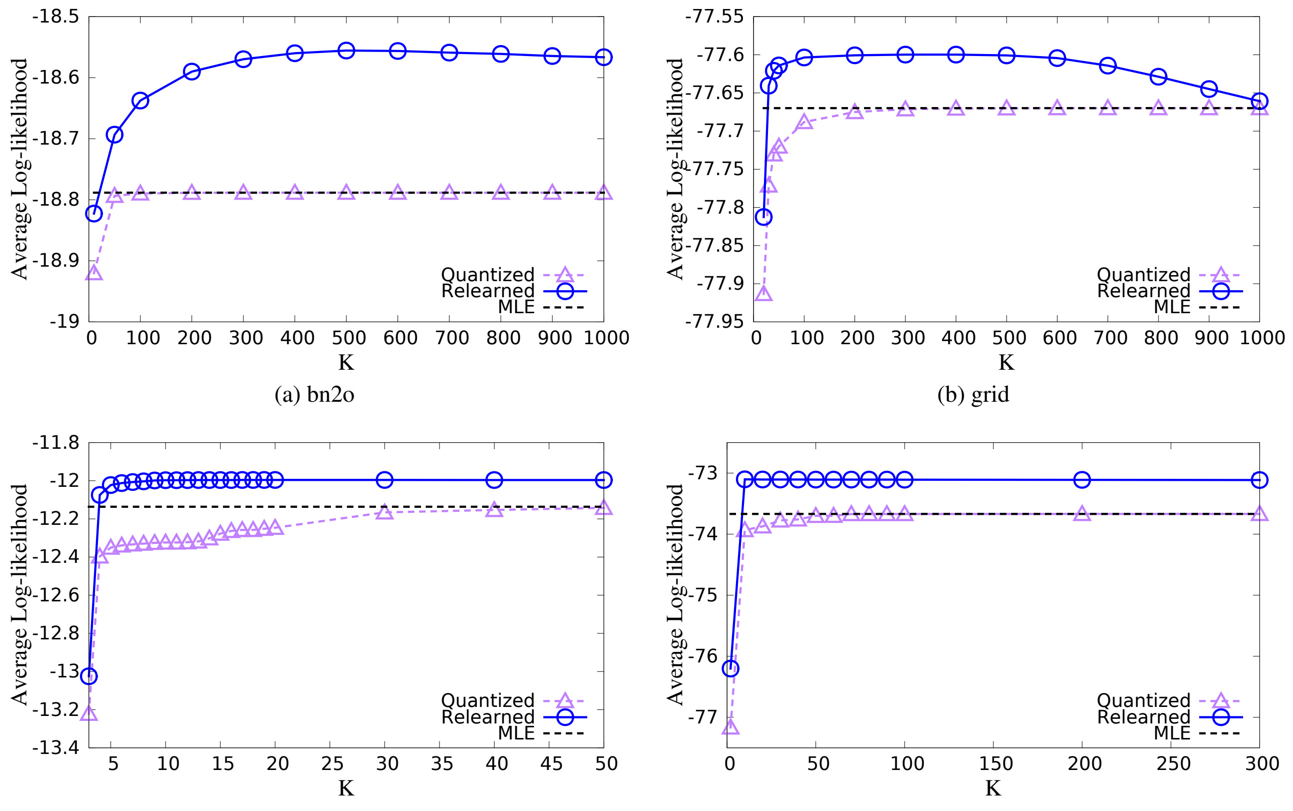
We experimented on various noisy-or networks. The results for these networks were consistently similar. Figure 2 (a) shows the result for one of the networks. For these models, the MLE has a higher average log-likelihood than the quantized MLE at low values for k, but we obtained a significant performance improvement over the MLE by relearning after quantization. As k increases, relearning greatly outperforms the MLE and reaches a steady level. The performance difference tends to zero as k converges to the actual number of parameter in the network. Even for small k, the relatively small difference in log-likelihood appears to be a reasonable trade-off for the drastic reduction in the number of model parameters.

We also selected two relational networks and one grid network. The results for these models appear in Figure 2 (b)-(d). The performance for these networks are similar to the noisy-or networks. For small k, the performance improvement by relearning is realized fairly immediately, while the quantization also converges to the MLE quickly.

We conducted error analysis on parameter estimation between the three different learned models and the true model by using the validation set to obtain an optimally estimated k. With the exception of bn2o, our estimated k is reasonably close to the true k in the various models as shown in third and fourth column of Table 1. The bn2o networks contains many very similar parameter values and thus resulted in a much lower estimated k. For each model set, we calculated the average absolute point-wise error between the true and learned parameters and selected the maximum among the experimental set. The results are also shown in Table 1. The relearning has a consistent lower error rate compared to MLE and quantization, while the quantization has similar error rate as MLE with fewer parameters. This reduction in the number of parameters often translates to better inference performance at prediction time as we show in the next subsection.

Inference

We compared our proposed approximate inference method based on slice importance sampling (denoted TW) to MC-SAT (Poon and Domingos 2006) in the Alchemy system (Kok et al. 2006) and Gibbs sampling for inference in Bayesian networks. Each algorithm was run for 500 seconds and then evaluated by computing the average Hellinger distance (Kokolakis and Nanopoulos 2001) between the



(a) bn2o (b) grid (c) students (d) friends

Figure 2: Average log-likelihood on test data plotted for each parameter learned graphical model (MLE, Quantized and Relearned) varying the value for k level of quantization.

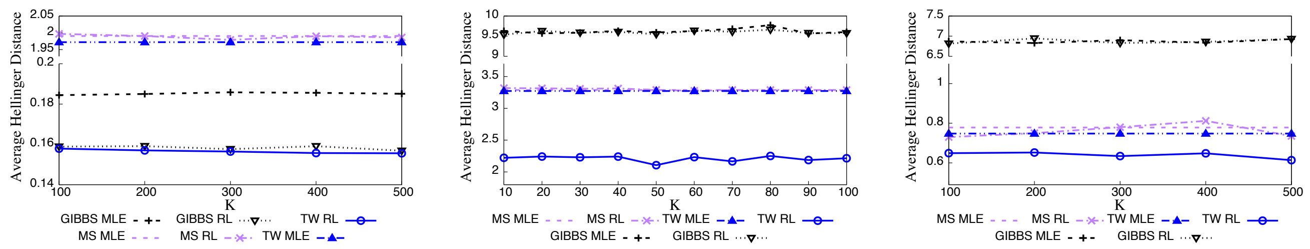
single-variable marginals obtained by each algorithm and marginals obtained by an exact solver. The results were averaged over 10 runs of each algorithm using the MLE as well as parameters that were relearned (RL) after quantization. Figure 3 (a)-(c) shows our experimental results on each the three types of networks (noisy-or, relational and grid).

The average Hellinger distances are consistently similar between MC-SAT and our method across the three network types when no parameters are tied (the MLE case). This is expected as MC-SAT is a special case of our algorithm. However, with the relearned parameters, our algorithm significantly outperforms both MC-SAT and Gibbs sampling on the students and grid networks. This shows that even though the test-set log likelihood of the MLE solution and the parameter tied models are roughly the same, at prediction time (estimating marginals), models having tied parameters outperform untied models provided methods such as TW that explicitly exploit the tied parameters are used.

One explanation for the poor performance of MC-SAT on parameter tied models is that MC-SAT is based on local search with a strong bias towards features that have high weights. Thus, without the ability to make large moves, MC-SAT is not able to efficiently traverse through the state space. Analogously, MCMC techniques such as Gibbs sampling, can also become trapped within a local region and may require a large number of samples to escape. Since our algorithm systematically partitions the overall state-space through the tied weight structure, it is able to move across the various regions more easily.

Discussion

We proposed a greedy method to learn tied parameter models that quantizes the parameters learned via maximum likelihood estimation using k-means clustering. Despite its simplicity, we demonstrated empirically that our approach can be used both as a regularizer and as a technique to reduce model complexity while maintaining predictive performance, which comports with the theoretical bounds on the error resulting from quantization that we provided. We also introduced a new importance sampling technique that exploits the symmetry resulting from the quantization in order to sample more effectively from tied parameter models than MC-SAT and Gibbs sampling. In future work, we plan to investigate applications of these techniques to Markov networks, bounds on the sample complexity required to obtain the correct quantization, and applications of quantization to structure learning.



(a) bn2o (b) students (c) grid

Figure 3: Average Hellinger distance between the exact and the approximate one-variable marginals plotted as a function of k level of quantization for MS MLE (MC-SAT MLE), MS RL (MC-SAT Relearned), TW MLE (Tied Weight MLE), TW RL (Tied Weight Relearned), GIBBS MLE (Gibbs MLE) and GIBBS RL (Gibbs Relearned). Result for each of the network types (noisy-or, relational and grid) are shown.

Acknowledgements

This work was supported in part by the DARPA Probabilistic Programming for Advanced Machine Learning Program under AFRL prime contract number FA8750-14-C-0005 and by the NSF award 1528037. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views or official policies, either expressed or implied, of DARPA, NSF, AFRL, ARO or the US government.

References

Baum, L. E.; Petrie, T.; Soules, G.; and Weiss, N. 1970. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics* 41(1):164–171.

Cheng, J., and Druzdzel, M. J. 2001. Confidence Inference in Bayesian Networks. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*, 75–82.

Fung, R. M., and Chang, K. 1989. Weighing and Integrating Evidence for Stochastic Simulation in Bayesian Networks. In *Proceedings of the Fifth Conference on Uncertainty in Artificial Intelligence*, 209–220.

Getoor, L., and Taskar, B., eds. 2007. *Introduction to Statistical Relational Learning*. MIT Press.

Gogate, V., and Dechter, R. 2011. SampleSearch: Importance Sampling in Presence of Determinism. *Artificial Intelligence* 175(2):694–729.

Gogate, V., and Domingos, P. 2010. Formula-Based Probabilistic Inference. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, 210–219.

Gogate, V. 2009. *Sampling Algorithms for Probabilistic Graphical Models with Determinism*. Ph.D. Dissertation, University of California, Irvine.

Jha, A.; Gogate, V.; Meliou, A.; and Suciu, D. 2010. Lifted Inference from the Other Side: The tractable Features. In *Proceedings of the 24th Annual Conference on Neural Information Processing Systems (NIPS)*, 973–981.

Kok, S.; Sumner, M.; Richardson, M.; Singla, P.; Poon, H.; and Domingos, P. 2006. The Alchemy System for Statistical Relational AI. Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA. http://alchemy.cs.washington.edu.

Kokolakis, G., and Nanopoulos, P. 2001. Bayesian Multivariate Micro-Aggregation Under the Hellinger's Distance Criterion. *Research in Official Statistics* 4:117–125.

Lafferty, J.; McCallum, A.; and Pereira, F. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, 282–289.

Liu, J. S. 2001. *Monte Carlo Strategies in Scientific Computing*. Springer Publishing Company, Incorporated.

Milch, B.; Zettlemoyer, L. S.; Kersting, K.; Haimes, M.; and Kaelbling, L. P. 2008. Lifted Probabilistic Inference with Counting Formulas. In *Proceedings of the Twenty-Third National Conference on Artificial Intelligence*, 1062–1068.

Murphy, K. 2002. *Dynamic Bayesian Networks: Representation, Inference and Learning*. Ph.D. Dissertation, UC Berkeley, Computer Science Division.

Neal, R. 2000. Slice Sampling. *Annals of Statistics* 31(3):705–767.

Ortiz, L. E., and Kaelbling, L. P. 2000. Adaptive Importance Sampling for Estimation in Structured Domains. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, 446–454.

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Poon, H., and Domingos, P. 2006. Sound and Efficient Inference with Probabilistic and Deterministic Dependencies. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 458–463.

Sang, T.; Beame, P.; and Kautz, H. 2005. Solving Bayesian Networks by Weighted Model Counting. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 475–482.

Savicky, P., and Vomlel, J. 2009. Triangulation Heuristics for BN2O Networks. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*. Springer Berlin Heidelberg. 566–577.

Wang, H., and Song, M. 2011. Ckmeans. 1d. dp: Optimal k-means Clustering in One Dimension by Dynamic Programming. *The R Journal* 3(2):29–33.

Wei, W.; Erenrich, J.; and Selman, B. 2004. Towards Efficient Sampling: Exploiting Random Walk Strategies. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence*, 670–676.