Tensor Belief Propagation

Andrew Wrigley [1] **Wee Sun Lee** [2] **Nan Ye** [3]

Abstract

We propose a new approximate inference algorithm for graphical models, *tensor belief propagation*, based on approximating the messages passed in the junction tree algorithm. Our algorithm represents the potential functions of the graphical model and all messages on the junction tree compactly as mixtures of rank-1 tensors. Using this representation, we show how to perform the operations required for inference on the junction tree efficiently: marginalisation can be computed quickly due to the factored form of rank-1 tensors while multiplication can be approximated using sampling. Our analysis gives sufficient conditions for the algorithm to perform well, including for the case of high-treewidth graphs, for which exact inference is intractable. We compare our algorithm experimentally with several approximate inference algorithms and show that it performs well.

1. Introduction

Probabilistic graphical models provide a general framework to conveniently build probabilistic models in a modular and compact way. They are commonly used in statistics, computer vision, natural language processing, machine learning and many related fields (Wainwright & Jordan, 2008). The success of graphical models depends largely on the availability of efficient inference algorithms. Unfortunately, exact inference is intractable in general, making approximate inference an important research topic.

Approximate inference algorithms generally adopt a variational approach or a sampling approach. The variational approach formulates the inference problem as an optimisation problem and constructs approximations by solving relaxations of the optimisation problem. A number of well-known inference algorithms can be seen as variational algorithms, such as loopy belief propagation, mean-field variational inference, and generalized belief propagation (Wainwright & Jordan, 2008). The sampling approach uses sampling to approximate either the underlying distribution or key quantities of interest. Commonly used sampling methods include particle filters and Markov-chain Monte Carlo (MCMC) algorithms (Andrieu et al., 2003).

Our proposed algorithm, *tensor belief propagation* (TBP), can be seen as a sampling-based algorithm. Unlike particle filters or MCMC methods, which sample states (also known as particles), TBP samples functions in the form of rank-1 tensors. Specifically, we use a data structure commonly used in exact inference, the junction tree, and perform approximate message passing on the junction tree using messages represented as mixtures of rank-1 tensors.

We assume that each factor in the graphical model is originally represented as a tensor decomposition (mixture of rank-1 tensors). Under this assumption, all messages and intermediate factors also have the same representation. Our key observation is that marginalisation can be performed efficiently for mixtures of rank-1 tensors, and multiplication can be approximated by sampling. This leads to an approximate message passing algorithm where messages and intermediate factors are approximated by low-rank tensors.

We provide analysis, giving conditions under which the method performs well. We compare TBP experimentally with several existing approximate inference methods using Ising models, random MRFs and two real-world datasets, demonstrating promising results.

2. Related Work

Exact inference on tree-structured graphical models can be performed efficiently using *belief propagation* (Pearl, 1982), a dynamic programming algorithm that involves passing *messages* between nodes containing the results of intermediate computations. For arbitrary graphical models, the well-known *junction tree algorithm* (Shafer & Shenoy, 1990; Lauritzen & Spiegelhalter, 1988; Jensen et al., 1990) is commonly used. The model is first compiled into a *junc-*

[1]Australian National University, Canberra, Australia. [2]National University of Singapore, Singapore. [3]Queensland University of Technology, Brisbane, Australia. Correspondence to: Andrew Wrigley <andrew.wrigley@anu.edu.au>, Wee Sun Lee <leews@comp.nus.edu.sg>, Nan Ye <n.ye@qut.edu.au>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

tion tree data structure and a similar message passing algorithm is then run over the junction tree. Unfortunately, for non-tree models the time and space complexity of the junction tree algorithm grows exponentially with a property of the graph called its *treewidth*. For high-treewidth graphical models, exact inference is intractable in general.

Our work approximates the messages passed in the junction tree algorithm to avoid the exponential runtime caused by exact computations in high-treewidth models. Various previous work has taken the same approach. Expectation propagation (EP) (Minka, 2001) approximates messages by minimizing the Kullback-Leiber (KL) divergence between the actual message and its approximation. Structured message passing (Gogate & Domingos, 2013) can be considered as a special case of EP where structured representations, in particular algebraic decision diagrams (ADDs) and sparse hash tables, are used so that EP can be performed efficiently. In contrast to ADDs, the tensor decompositions used for TBP may provide a more compact representation for some problems. An ADD partitions a tensor into axis-aligned hyper-rectangles – it is possible to represent a hyper-rectangle using a rank-1 tensor but a rank-1 tensor is generally not representable as an axis-aligned rectangle. Furthermore, the supports of the rank-1 tensors in the mixture may overlap. However, ADD compresses hyper-rectangles that share sub-structures and this may result in the methods having different strengths. Sparse tables, on the other hand, work well for cases with extreme sparsity.

Several methods use sampled particles to approximate messages (Koller et al., 1999; Ihler & McAllester, 2009; Sudderth et al., 2010). To allow their algorithms to work well on problems with less sparsity, Koller et al. (1999) and Sudderth et al. (2010) use non-parametric methods to smooth the particle representation of messages. In contrast, we decompose each tensor into a mixture of rank-1 tensors and sample the rank-1 tensors directly, instead of through the intermediate step of sampling particles. Another approach, which pre-samples the particles at each node and passes messages between these pre-sampled particles was taken by Ihler & McAllester (2009). The methods of Ihler & McAllester (2009) and Sudderth et al. (2010) were also applied on graphs with loops using loopy belief propagation.

Xue et al. (2016) use discrete Fourier representations for inference via the elimination algorithm. The discrete Fourier representation is a special type of tensor decomposition. Instead of sampling, the authors perform approximations by truncating the Fourier coefficients, giving different approximation properties. Other related works include (Darwiche, 2000; Park & Darwiche, 2002; Chavira & Darwiche, 2005), where belief networks are compiled into compact arithmetic circuits (ACs). On the related problem of MAP inference, McAuley & Caetano (2011) show that junction tree clusters that factor over subcliques or consist only of latent variables yield improved complexity properties.

3. Preliminaries

For simplicity we limit our discussion to Markov random fields (MRFs), but our results apply equally to Bayesian networks and general factor graphs. We focus only on discrete models. A Markov random field G is an undirected graph representing a probability distribution $P(X_1, \ldots, X_N)$, such that P factorises over the max-cliques in G, i.e.

$$P(X_1, \ldots, X_N) = \frac{1}{Z} \prod_{c \in \mathrm{cl}(G)} \phi_c(\mathbf{X}_c) \tag{1}$$

where $\mathrm{cl}(G)$ is the set of max-cliques in G and $Z = \sum_{\mathbf{X}} \prod_{c \in \mathrm{cl}(G)} \phi_c(\mathbf{X}_c)$ ensures normalisation. We call the factors ϕ_c *clique potentials* or potentials.

TBP is based on the junction tree algorithm (see e.g. (Koller & Friedman, 2009)). A *junction tree* is a special type of cluster graph, i.e. an undirected graph with nodes called *clusters* that are associated with sets of variables rather than single variables. Specifically, a junction tree is a cluster graph that is a tree and which also satisfies the *running intersection property*. The running intersection property states that if a variable is in two clusters, it must also be in every cluster on the path that connects the two clusters.

The junction tree algorithm is essentially the well-known belief propagation algorithm applied to the junction tree after the cluster potentials have been initialized. At initialisation, each clique potential is first associated with a cluster. Each cluster potential $\Phi_t(\mathbf{X}_t)$ is computed by multiplying all the clique potentials $\phi_c(\mathbf{X}_c)$ associated with the cluster $\mathbf{X}_t$. Thereafter, the algorithm is defined recursively in terms of *messages* passed between neighbouring clusters. A message is always a function of the variables in the receiving cluster, and represents an intermediate marginalisation over a partial set of factors. The message $m_{t \to s}(\mathbf{X}_s)$ sent from a cluster t to a neighbouring cluster s is defined recursively by

$$m_{t \to s}(\mathbf{X}_s) = \sum_{\mathbf{X}_t \setminus \mathbf{X}_s} \Phi_t(\mathbf{X}_t) \prod_{u \in N(t) \setminus \{s\}} m_{u \to t}(\mathbf{X}_t), \tag{2}$$

where $N(t)$ is the set of neighbours of t. Since the junction tree is singly connected, this recursion is well-defined. After all messages have been computed, the marginal distribution on a cluster of variables $\mathbf{X}_s$ is computed using

$$P_s(\mathbf{X}_s) \propto \Phi_s(\mathbf{X}_s) \prod_{t \in N(s)} m_{t \to s}(\mathbf{X}_s), \tag{3}$$

and univariate marginals can be computed by summation over cluster marginals. The space and time complexity of

the junction tree inference algorithm is exponential in the *induced width* of the graph, i.e. the number of variables in the largest tree cluster minus 1 (Koller & Friedman, 2009). The lowest possible induced width (over all possible junction trees for the graph) is defined as the *treewidth* of the graph[1].

4. Tensor Belief Propagation

The TBP algorithm we propose (Algorithm 1) is the same as the junction tree algorithm except for the approximations at line 1 and line 4, which we describe below.

Algorithm 1 Tensor Belief Propagation

input Clique potentials $\{\phi_c(\mathbf{X}_c)\}$, junction tree J.
output Approximate cluster marginals $\{\tilde{P}_s(\mathbf{X}_s)\}$ for all s.

1: For each cluster $\mathbf{X}_t$, compute $\tilde{\Phi}_t(\mathbf{X}_t) \approx \prod_c \phi_c(\mathbf{X}_c)$, where the product is over all cliques c associated with t.
2: **while** there is any unsent message **do**
3: Pick an unsent message $m_{t\to s}(\mathbf{X}_s)$ with all messages to t from neighbours other than s sent.
4: Send $\tilde{m}_{t\to s}(\mathbf{X}_s) \approx \sum_{\mathbf{X}_t \setminus \mathbf{X}_s} \tilde{\Phi}_t(\mathbf{X}_t) \prod_{u\in N(t)\setminus\{s\}} \tilde{m}_{u\to t}(\mathbf{X}_t)$.
5: **end while**
6: **return** $\tilde{P}_s(\mathbf{X}_s) \propto \tilde{\Phi}_s(\mathbf{X}_s) \prod_{t\in N(s)} \tilde{m}_{t\to s}(\mathbf{X}_s)$.

There are two challenges in applying the junction tree algorithm to high-treewidth graphical models: representing the intermediate potential functions, and computing them. For representation, using tables to represent the cluster potentials and messages requires space exponential in the induced width. For computation, the two operations relevant to the complexity of the algorithm are marginalisation over a subset of variables in a cluster and multiplication of multiple factors. When clusters become large, these operations become intractable unless additional structure can be exploited.

TBP alleviates these difficulties by representing all potential functions as mixtures of rank-1 tensors. We show how to perform exact marginalisation of a mixture (required in line 4) efficiently, and how to perform approximate multiplication of mixtures using sampling (used for approximation in lines 1 and 4).

4.1. Mixture of Rank-1 Tensors

As we are concerned with discrete distributions, each potential ϕ_c can be represented by a multidimensional array, i.e. a *tensor*. Furthermore, a d-dimensional tensor $\mathbf{T} \in \mathbb{R}^{N_1\times\cdots\times N_d}$ can be decomposed into a weighted sum of outer products of vectors as

$$\mathbf{T} = \sum_{k=1}^{r} w_k \, \mathbf{a}_k^1 \otimes \mathbf{a}_k^2 \otimes \cdots \otimes \mathbf{a}_k^d, \tag{4}$$

where $w_k \in \mathbb{R}$, $\mathbf{a}_k^i \in \mathbb{R}^{N_i}$ and $\otimes$ is the outer product, i.e. $\left(\mathbf{a}_k^1 \otimes \mathbf{a}_k^2 \otimes \cdots \otimes \mathbf{a}_k^d\right)_{i_1,\ldots,i_d} = \left(\mathbf{a}_k^1\right)_{i_1} \cdot \cdots \cdot \left(\mathbf{a}_k^d\right)_{i_d}$. We denote the vector of weights $\{w_k\}$ as $\mathbf{w}$. The smallest r for which an exact r-term decomposition exists is called the *rank* of $\mathbf{T}$ and a decomposition (4) using this r is a *tensor rank decomposition*. This decomposition is known by several names including CANDECOMP/PARAFAC (CP) decomposition and Hitchcock decomposition (Kolda & Bader, 2009). In this paper, we assume without loss of generality that the weights are non-negative and sum to 1, giving a mixture of rank-1 tensors. Such a mixture forms a probability distribution over rank-1 tensors, which we refer to as a *tensor belief*. We also assume that the decomposition is non-negative, i.e. the rank-1 tensors are non-negative, although the method can be extended to allow negative values.

For a (clique or cluster) potential function $\phi_s(\mathbf{X}_s)$ over $|\mathbf{X}_s| = N_s$ variables, (4) is equivalent to decomposing ϕ_s into a sum of fully-factorised terms, i.e.

$$\begin{aligned}\phi_s(\mathbf{X}_s) &= \sum_{k=1}^{r} w_k^s \, \psi_k^s(\mathbf{X}_s) \\ &= \sum_{k=1}^{r} w_k^s \, \psi_{k,1}^s(X_{s_1}) \cdots \psi_{k,N_s}^s(X_{s_{N_s}}).\end{aligned} \tag{5}$$

This observation allows us to perform marginalisation and multiplication operations efficiently.

4.2. Marginalisation

Marginalising out a variable X_i from a cluster $\mathbf{X}_s$ simplifies in the same manner as if the distribution was fully-factorised, namely

$$\sum_{X_i} \phi_s(\mathbf{X}_s) = \sum_{k=1}^{r} w_k^s \left(\sum_{X_i} \psi_{k,j}^s(X_i)\right) \cdot \underbrace{\psi_{k,1}^s(X_{s_1})\psi_{k,2}^s(X_{s_2})\cdots\psi_{k,N_s}^s(X_{s_{N_s}})}_{\text{excluding } \psi_{k,j}^s(X_i)} \tag{6}$$

where we simply push the sum $\sum_{X_i}$ inside and only evaluate it over the univariate factor $\psi_{k,j}^s(X_i)$. We can then absorb this sum into the weights $\{w_k^s\}$ and the result stays in decomposed form (5). To marginalise over multiple variables, we evaluate (6) for each variable in turn.

4.3. Multiplication

The key observation for multiplication is that a mixture of rank-1 tensors can be treated as a probability distribution

[1] Finding the treewidth of a graph is $\mathcal{NP}$-hard; in practice we use various heuristics to construct the junction tree.

over the rank-1 tensors with expectation equal to the true function, by considering the weight of each rank-1 term w_k as its probability.

To multiply two potentials ϕ_i and ϕ_j, we repeatedly sample rank-1 terms from each multiplicand to build the product. We draw a sample of K pairs of indices $\{(k_r, l_r)\}_{r=1}^{K}$ independently from $\mathbf{w}^i$ and $\mathbf{w}^j$ respectively, and use the approximation

$$\phi_i(\mathbf{X}_i) \cdot \phi_j(\mathbf{X}_j) \approx \frac{1}{K} \sum_{r=1}^{K} \psi_{k_r}^i(\mathbf{X}_i) \cdot \psi_{l_r}^j(\mathbf{X}_j). \quad (7)$$

The approximation is also a mixture of rank-1 tensors, with the rank-1 tensors being the $\psi_{k_r}^i(\mathbf{X}_i) \cdot \psi_{l_r}^j(\mathbf{X}_j)$, and their weights being the frequencies of the sampled (k_r, l_r) pairs. This process is equivalent to drawing a sample of the same size from the distribution representing $\phi_i(\mathbf{X}_i) \cdot \phi_j(\mathbf{X}_j)$ and hence provides an unbiased estimate of the product function. Multiplication of each pair $\psi_{k_r}^i(\mathbf{X}_i) \cdot \psi_{l_r}^j(\mathbf{X}_j)$ can be performed efficiently due to their factored forms. It is possible to extend the method to allow multiplication of more potential functions simultaneously but we only use pairwise multiplication in this paper, repeating the process as necessary to multiply more functions.

4.4. Theoretical Analysis

For simplicity, we give results for binary MRFs. Extension to non-binary variables is straightforward. We assume that exact tensor decompositions are given for all initial clique potentials.

Theorem 1. Consider a binary MRF with C max-cliques with potential functions represented as non-negative tensor decompositions. Consider the unnormalised distribution $D(\mathbf{X}) = \prod_{c \in \mathrm{cl}(G)} \phi_c(\mathbf{X}_c)$. Let $D_i(X_i) = \sum_{\mathbf{X} \setminus X_i} D(\mathbf{X})$ be the unnormalised marginal for variable X_i. Assume that the values of the rank-1 tensors in any mixture resulting from pairwise multiplication is upper bounded by M, and that the approximation target for any cell in any multiplication operation is lower bounded by B. Let $\tilde{D}_i(X_i)$ be the estimates produced by TBP using a junction tree with an induced width T. With probability at least $1 - \delta$, for all i and x_i,

$$(1 - \epsilon) D_i(x_i) \leq \tilde{D}_i(x_i) \leq (1 + \epsilon) D_i(x_i)$$

if the sample size used for all multiplication operations is at least

$$K_{\min}(\epsilon, \delta) \in \mathcal{O}\left(\frac{C^2}{\epsilon^2} \frac{M}{B} \left(\log C + T + \log \frac{2}{\delta}\right)\right).$$

Furthermore, $\tilde{D}_i(X_i)$ remains a consistent estimator for $D_i(X_i)$ if $B = 0$.

Proof. We give the proof for the case $B \neq 0$ here. The consistency proof for the case $B = 0$ can be found in the supplementary material.

The errors in the unnormalised marginals are due to the errors in approximate pairwise multiplication defined by Equation (7). We first give bounds for the errors in all the pairwise multiplication by sampling operations, then derive the bounds for the errors in the unnormalised marginals.

Recall that by the multiplicative Chernoff bound (see e.g. (Koller & Friedman, 2009)), for K i.i.d. random variables $Y_1, \ldots, Y_K$ in range $[0, M]$ with expected value μ, we have

$$P\left(\frac{1}{K} \sum_{i=1}^{K} Y_i \notin [(1 - \zeta)\mu, (1 + \zeta)\mu]\right) \leq 2 \exp\left(-\frac{\zeta^2 \mu K}{3M}\right).$$

The number of pairwise multiplications required to initialize the cluster potentials $\{\tilde{\Phi}_t(\mathbf{X}_t)\}$ is at most C (line 1). The junction tree has at most C clusters, and thus at most $2(C - 1)$ messages need to be computed. Each message requires at most C pairwise multiplications (line 4). Hence the total number of pairwise multiplications needed is at most $2C^2$. Each pairwise multiplication involves functions defined on at most T binary variables, and thus it simultaneously estimates at most 2^T values. Hence at most $2^{T+1}C^2$ values are estimated by sampling in the algorithm.

Since each estimate satisfies the Chernoff bound, we can apply a union bound to bound the probability that the estimate for any μ_j is outside $[(1 - \zeta)\mu_j, (1 + \zeta)\mu_j]$, giving

$$P_\zeta\left(\frac{1}{K} \sum_{i=1}^{K} X_i^j \notin [(1 - \zeta)\mu_j, (1 + \zeta)\mu_j] \text{ for any estimate } j\right) \leq 2^{T+2} C^2 \exp\left(-\frac{\zeta^2 B K}{3M}\right),$$

where B is a lower bound on the minimum value of all cells estimated during the algorithm. If we set an upper-bound on this error probability of δ and rearrange for K, we have that with probability at least δ, all estimates are within a factor of $(1 \pm \zeta)$ of their true values when

$$K \geq \frac{3}{\zeta^2} \frac{M}{B} \ln \frac{2^{T+2} C^2}{\delta}. \quad (8)$$

We now seek an expression for the sample size required for the unnormalised marginals to have small errors. First we argue that at most $C + Q - 1$ pairwise multiplications are used in the process of constructing the marginals at any node u, where Q is the number of clusters. This is true for the base case of a tree of size 1 as no messages need to be passed. As the inductive hypothesis, assume that the statement is true for trees of size less than n. The node u is

considered as the root of the tree and by the inductive hypothesis the number of multiplications used in each subtree i is at most $C_i + Q_i - 1$ where C_i and Q_i are the number of clique potentials and clusters associated with subtree i. Summing them up and noting that we need to perform at most one additional multiplication for each clique potential associated with u for initialisation, and one additional multiplication for each subtree, gives us the required result. To simplify analysis, we bound $C + Q - 1$ by $2C$ from here onwards.

At worst, each pairwise multiplication results in an extra $(1 \pm \zeta)$ factor in the bound. Since we are using a multiplicative bound, marginalisation operations have no effect on the bound. As we do no more than $2C$ multiplications, the final marginal estimates are all within a factor $(1\pm\zeta)^{2C}$ of their true value.

To bound the marginal so that it is within a factor $(1 \pm \epsilon)$ of its true value for a chosen $\epsilon > 0$, we note that choosing $\zeta = \frac{\ln(1+\epsilon)}{2C}$ implies $(1-\zeta)^{2C} \geq 1 - \epsilon$ and $(1+\zeta)^{2C} \leq (1+\epsilon)$:

$$(1-\zeta)^{2C} = \left(1 - \frac{\ln(1+\epsilon)}{2C}\right)^{2C} \tag{9}$$
$$\geq \left(1 - \frac{\epsilon}{2C}\right)^{2C} \geq 1 - \epsilon$$
$$(1+\zeta)^{2C} = \left(1 + \frac{\ln(1+\epsilon)}{2C}\right)^{2C} \tag{10}$$
$$\leq \exp\left(\ln(1+\epsilon)\right) = 1 + \epsilon.$$

In (9) we use Bernoulli's inequality together with $\ln(1+\epsilon) \leq \epsilon$, and in (10) we use $(1+x)^r \leq \exp(rx)$.

Substituting this ζ into (8), we have that all marginal estimates are accurate within factor $(1 \pm \epsilon)$ with probability at least $1 - \delta$, when

$$K \geq \frac{12C^2}{(\ln(1+\epsilon))^2} \frac{M}{B} \ln \frac{2^{T+2}C^2}{\delta} \tag{11}$$

Using the fact that $\ln(1+\epsilon) \geq \epsilon \cdot \ln 2$ for $0 \leq \epsilon \leq 1$, and $\frac{24}{\ln 2} < 35$, we can relax this bound to

$$K \geq \frac{35C^2}{\epsilon^2} \frac{M}{B} \left(\log C + T + \log \frac{2}{\delta}\right).$$

□

Corollary 1. Under the same conditions as Theorem 1, with probability at least $1 - \delta$, the normalised marginal estimates $\tilde{p}_i(x_i)$ satisfy

$$(1-\gamma)p_i(x_i) \leq \tilde{p}_i(x_i) \leq (1+\gamma)p_i(x_i)$$

for all i and x_i, if the sample size used for all multiplication operations is at least

$$K'_{\min}(\gamma, \delta) \in \mathcal{O}\left(\frac{C^2}{\gamma^2} \frac{M}{B} \left(\log C + T + \log \frac{1}{\delta}\right)\right).$$

Proof. Suppose the unnormalised marginals have relative error bounded by $(1 \pm \epsilon)$, i.e.

$$(1-\epsilon)D_i(x_i) \leq \tilde{D}_i(x_i) \leq (1+\epsilon)D_i(x_i)$$

for all i and x_i. Then we have

$$\tilde{p}_i(x_i) = \frac{\tilde{D}_i(x_i)}{\sum_{x_i} \tilde{D}_i(x_i)} \leq \frac{(1+\epsilon)D_i(x_i)}{\sum_{x_i}(1-\epsilon)D_i(x_i)} = \frac{1+\epsilon}{1-\epsilon} p_i(x_i).$$

To bound the relative error on $\tilde{p}_i(x_i)$ to $(1+\gamma)$, we set

$$\frac{1+\epsilon}{1-\epsilon} = 1 + \gamma \implies \epsilon = \frac{\gamma}{\gamma+2}.$$

Since $\frac{1}{\epsilon^2} = \left(\frac{\gamma+2}{\gamma}\right)^2 < \left(\frac{3}{\gamma}\right)^2 = \mathcal{O}\left(\frac{1}{\gamma^2}\right)$ for $\gamma < 1$, the increase in $K_{\min}$ required to bound the normalised estimates rather than the unnormalised estimates is at most a constant. Thus,

$$K'_{\min}(\gamma, \delta) \in \mathcal{O}\left(\frac{C^2}{\gamma^2} \frac{M}{B} \left(\log C + T + \log \frac{1}{\delta}\right)\right).$$

as required. The negative side of the bound is similar. □

Interestingly, the sample size does not grow exponentially with the induced width, and hence the treewidth of the graph. As the inference problem is $\mathcal{NP}$-hard, we expect the ratio M/B to be large in difficult problems. The M/B ratio comes from bounding the relative error when sampling a mixture $\phi(\mathbf{x}) = \sum_{k=1}^{r} w_k \psi_k(\mathbf{x})$. A more refined bound can be obtained by bounding $\max_k \max_{\mathbf{x}} \psi_k(\mathbf{x})/\phi(\mathbf{x})$ instead; this bound would not grow as quickly if $\psi_k(\mathbf{x})$ is always small whenever $\phi(\mathbf{x})$ is small. Understanding the properties of function classes where these bounds are small may help us understand when TBP works well.

Theorem 1 suggests that that it may be useful to reweight the rank-1 tensors in a multiplicand to give a smaller M/B ratio. The following proposition gives a reweighting scheme that minimises the maximum value of the rank-1 tensors in a multiplicand, which leads to a smaller M with B fixed. Theorem 1 still holds with this reweighting.

Proposition 1. Let $\phi(\mathbf{x}) = \sum_{k=1}^{r} w_k \psi_k(\mathbf{x})$ where $w_k \geq 0$, $\sum_{k=1}^{r} w_k = 1$ and $\psi_k(\mathbf{x}) \geq 0$ for all $\mathbf{x}$. Consider a reweighted representation $\sum_{k=1}^{r} w'_k \psi'_k(\mathbf{x})$, where $\psi'_k(\mathbf{x}) = \frac{w_k}{w'_k}\psi_k(\mathbf{x})$, each $w'_k \geq 0$, and $\sum_k w'_k = 1$. Then $\max_k \max_{\mathbf{x}} \psi'_k(\mathbf{x})$ is minimized when $w'_k \propto w_k \max_{\mathbf{x}} \psi_k(\mathbf{x})$.

Proof. Let $a_k = w_k \max_{\mathbf{x}} \psi_k(\mathbf{x})$, and let $v = \max_k \max_{\mathbf{x}} \psi'_k(\mathbf{x}) = \max_k \frac{a_k}{w'_k}$. For any choice of w', we have $v \geq \frac{\sum_k a_k}{\sum_k w'_k} = \sum_k a_k$. The first inequality holds because $v w'_k \geq a_k$ for any k by the definition of v. Since

$\sum_k a_k$ is a constant lower bound for v, and this is clearly achieved by setting $w'_k \propto a_k$, we have the claimed result. □

Note that with $\psi_k(\mathbf{x})$ represented as a rank-1 tensor, the maximum value over $\mathbf{x}$ can be computed quickly with the help of the factored structure.

Reweighting the rank-1 tensors as described in Proposition 1 and then sampling gives a form of importance sampling. Importance sampling is often formulated instead with the objective of minimizing the variance of the estimates. Since we expect lowering the variance of intermediate estimates to lead to lower variance on the final marginal estimates, we also examine the following alternative reweighting scheme.

Proposition 2. Let $\phi(\mathbf{x}) = \sum_{k=1}^{r} w_k \psi_k(\mathbf{x})$ where $w_k \geq 0$, $\sum_{k=1}^{r} w_k = 1$. Consider a reweighted representation $\sum_{k=1}^{r} w'_k \psi'_k(\mathbf{x})$, where $\psi'_k(\mathbf{x}) = \frac{w_k}{w'_k}\psi_k(\mathbf{x})$, each $w'_k \geq 0$, and $\sum_k w'_k = 1$. Let $\tilde{\phi}(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K} \psi'_{\mathcal{K}_i}(\mathbf{x})$ be an estimator for $\phi(\mathbf{x})$, where each $\mathcal{K}_i$ is drawn independently from the categorical distribution with parameters $\{w'_1, \ldots, w'_r\}$. Then $\tilde{\phi}(\mathbf{x})$ is unbiased, and the total variance $\sum_{\mathbf{x}} \mathrm{Var}[\tilde{\phi}'(\mathbf{x})]$ is minimized when $w'_k \propto w_k \sqrt{\sum_{\mathbf{x}} \psi_k(\mathbf{x})^2}$.

Proof. Clearly $\tilde{\phi}(\mathbf{x})$ is an unbiased estimator for $\phi(\mathbf{x})$. The variance of this estimator is

$$\begin{aligned}
\mathrm{Var}[\tilde{\phi}(\mathbf{x})] &= \frac{1}{K}\mathrm{Var}[\psi'_{\mathcal{K}'_i}(\mathbf{x})] \\
&= \frac{1}{K}\left(\mathbb{E}[\psi'_{\mathcal{K}_i}(\mathbf{x})^2] - \mathbb{E}[\psi'_{\mathcal{K}_i}(\mathbf{x})]^2\right) \\
&= \frac{1}{K}\left(\sum_k w'_k \psi'_k(\mathbf{x})^2 - \phi(\mathbf{x})^2\right) \\
&= \frac{1}{K}\left(\sum_k w'_k \frac{w_k^2}{w'^2_k}\psi_k(\mathbf{x})^2 - \phi(\mathbf{x})^2\right) \\
&= \frac{1}{K}\left(\sum_k \frac{w_k^2}{w'_k}\psi_k(\mathbf{x})^2 - \phi(\mathbf{x})^2\right).
\end{aligned}$$

Since K and $\phi(\mathbf{x})$ are constant, we have

$$\begin{aligned}
\{w'_k\}^* &= \operatorname*{argmin}_{\{w'_k\}} \sum_{\mathbf{x}} \mathrm{Var}[\tilde{\phi}'(\mathbf{x})] \\
&= \operatorname*{argmin}_{\{w'_k\}} \sum_{\mathbf{x}} \sum_k \frac{1}{w'_k}(w_k \psi_k(\mathbf{x}))^2,
\end{aligned}$$

where each $w'_k \geq 0$, $\sum_k w'_k = 1$. A straightforward application of the method of Lagrange multipliers yields

$$w'_k = \frac{w_k \sqrt{\sum_{\mathbf{x}} \psi_k(\mathbf{x})^2}}{\sum_l w_l \sqrt{\sum_{\mathbf{x}} \psi_l(\mathbf{x})^2}}.$$

□

The factored forms of rank-1 tensors again allows the importance reweighting to be computed efficiently.

Propositions 1 and 2 could be applied directly to the algorithm by multiplying two potential functions fully before sampling. If each potential function has K terms, the multiplication would result in K^2 terms which is computationally expensive. We only partially exploit the results of Propositions 1 and 2 in our algorithm. When multiplying two potential functions, we draw K pairs of rank-1 tensors from their respective distributions and multiply them as described earlier but re-weight the resulting mixture using either Proposition 1 or 2. We call the former max-norm reweighting, and the latter min-variance reweighting.

5. Experiments

We present experiments on grid-structured Ising models, random graphs with pairwise Ising potentials, and two real-world datasets from the UAI 2014 Inference Competition (Gogate, 2014). We test our algorithm against commonly used approximate inference methods, namely loopy belief propagation (labelled BP), mean-field (MF), tree-reweighted BP (TRW) and Gibbs sampling using existing implementations from the libDAI package (Mooij, 2010). TBP was implemented in C++ inside the libDAI framework using Eigen (Guennebaud et al., 2010) and all tests were executed on a single core of a 1.4 GHz Intel Core i5 processor. In each experiment, we time the execution of TBP and allow Gibbs sampling the same wall-clock time. We run the other algorithms until convergence, which occurs quickly in each case (hence only the final performance is shown). Parameters used for BP, MF, TRW and Gibbs are given in the supplementary material. To build the junction tree, we use the min-fill heuristic[2] implemented in libDAI.

5.1. Grid-structured Ising Models

Figure 1 gives results for 10×10 Ising models. The $N \times N$ Ising model is a planar grid-structured MRF described by the joint distribution

$$p(x_1, \ldots, x_{N^2}) = \frac{1}{Z}\exp\left(\sum_{(i,j)} w_{ij} x_i x_j + \sum_i b_i x_i\right)$$

where (i, j) runs over all edges in the grid. Each variable X_i takes value either 1 or -1. In our experiments, we choose the w_{ij} uniformly from $[-2, 2]$ (mixed interactions) or $[0, 2]$ (attractive interactions), and the b uniformly from $[-1, 1]$. We use a symmetric rank-2 tensor decomposition for the pairwise potentials. We measure performance

[2] Min-fill repeatedly eliminates variables that result in the lowest number of additional edges in the graph (Koller & Friedman, 2009).

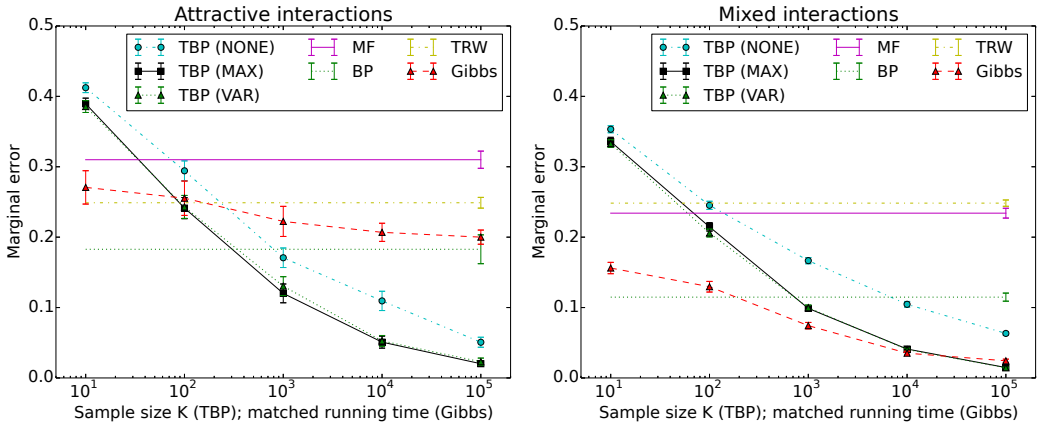


Figure 1: Marginal error of TBP for various sample sizes K on 10×10 Ising models compared with other approximate inference methods (NONE = no reweighting, MAX = max-norm reweighting, VAR = min-variance reweighting).

by the mean L_1 error of marginal estimates over the N^2 variables,

$$\frac{1}{N^2}\sum_{i=1}^{N^2}|P_{\text{exact}}(X_i=1)-P_{\text{approx}}(X_i=1)|.$$

Exact marginals were obtained using the exact junction tree algorithm (possible because the grid size is small). Results are all averages over 100 model instances generated randomly with the specified parameters; error bars show standard error. We give a description of the tensor decomposition and additional results for a range of grid sizes N and interaction strengths w_{ij} in the supplementary material.

As we increase the number of samples used for multiplication K, and hence running time, the performance of TBP improves as expected. On both attractive and mixed interaction models, loopy BP, mean-field and tree-reweighted BP perform poorly. Gibbs sampling performs poorly on models with attractive interactions but performs better on models with mixed interactions. This is expected since models with mixed interactions mix faster. Reweighting gives a noticeable improvement over not using reweighting; both max-norm reweighting and min-variance reweighting give very similar results (in Figure 1, they not easily differentiated). For the remainder of our experiments, we show results for max-norm reweighting only.

5.2. Random Pairwise MRFs

We also test TBP on random binary N-node MRFs with pairwise Ising potentials. We construct the graphs by independently adding each edge (i, j) with probability 0.5, with the requirement that the resulting graph is connected. Pairwise potentials are of the same form as Ising models (i.e. $\exp[w_{ij}x_ix_j]$), where interaction strengths w_{ij} are chosen uniformly from $[0, 2]$ (attractive) or $[-2, 2]$ (mixed) and the b_i are chosen uniformly from $[-1, 1]$.

Figure 2a shows the performance of TBP with increasing sample size K on 15-node models. Similar to grid-structured Ising models, TBP performs well as the sample size is increased. Notably, TBP performs very well on attractive models where other methods perform poorly.

Figure 2b shows the performance of the algorithms as the graph size is increased to 30 nodes. Interestingly, TBP starts to fail for mixed interaction models as the graph size is increased but remains very effective for attractive interaction models while other methods perform poorly.

5.3. UAI Competition Problems

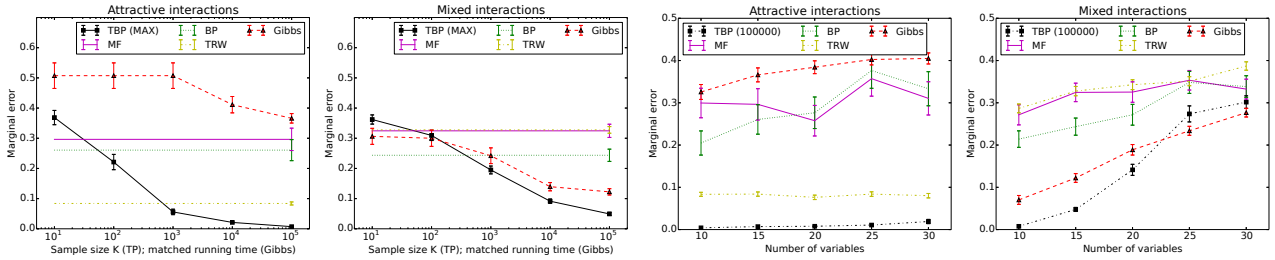
Finally, we show results of TBP on two real-world datasets from the UAI 2014 Inference Competition, namely the Promedus and Linkage datasets. We chose these two problems following Zhu & Ermon (2015). To compute the initial tensor rank decompositions, we use the non-negative `cp_nmu` method from Bader et al. (2015), an iterative optimisation method based on (Lee & Seung, 2001). The initial potential functions are decomposed into mixtures with r components. We show results for $r = 2$ and $r = 4$. We measure performance by the mean L_1 error over all states and all variables

$$\frac{1}{NS_i}\sum_{i=1}^{N}\sum_{x_i}|P_{\text{exact}}(X_i=x_i)-P_{\text{approx}}(X_i=x_i)|,$$

where S_i is the number of states of variable X_i. Results are averages over all problem instances in (Gogate, 2014).

We see that TBP outperforms Gibbs sampling on both problems[3]. On the Linkage dataset, using $r = 2$ mix-

[3]We omit results for BP, MF and TRW for these problems because the libDAI package was unable to run them.



(a) Effect of sample size K for $N = 15$ nodes

(b) Effect of graph size N

Figure 2: Performance of TBP on random pairwise MRFs compared with other approximate inference algorithms.

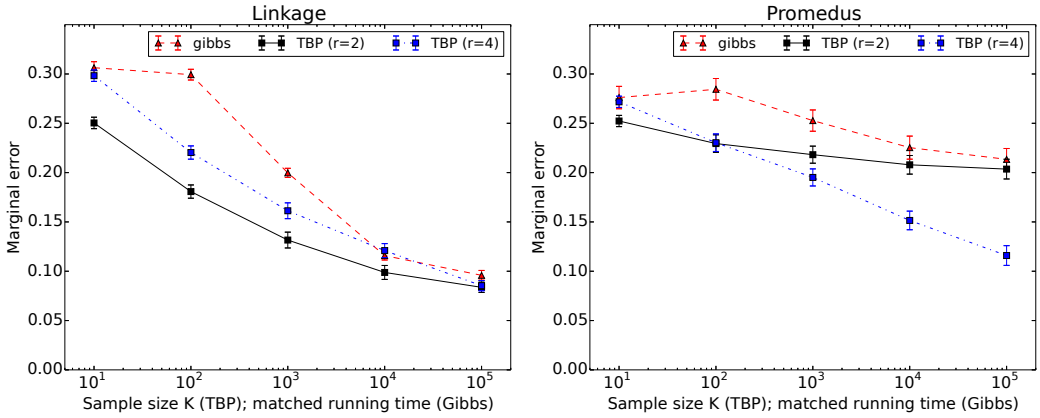


Figure 3: Marginal error of TBP vs Gibbs sampling on UAI 2014 Inference Competition problems for various sample sizes K.

ture components for the initial decompositions performs best and increasing r does not improve performance. On the Promedus dataset $r = 4$ performs better; in this case, increasing r may improve the accuracy of the decomposition of the initial potential functions. For reference, the marginal error achieved by the random projection method of Zhu & Ermon (2015) is 0.09 ± 0.02 for Linkage and 0.21 ± 0.06 for Promedus. TBP with $K = 10^5$ achieved error of 0.08 ± 0.01 for Linkage and 0.12 ± 0.01 for Promedus despite using substantially less running time [4].

6. Conclusion and Future Work

We proposed a new sampling-based approximate inference algorithm, tensor belief propagation, which uses a mixture-of-rank-1-tensors representation to approximate cluster potentials and messages in the junction tree algorithm. It gives consistent estimators, and performs well on a range of problems against well-known algorithms.

In this paper, we have not addressed how to perform the initial tensor decomposition. There exist well-known optimisation algorithms for this problem to minimize Euclidean error (Kolda & Bader, 2009), though it is not clear that this is the best objective for the purpose of TBP. We are currently investigating the best way of formulating and solving this optimisation problem to yield accurate results during inference. Further, when constructing the junction tree, it is not clear whether min-fill and other well-known heuristics remain appropriate for TBP. In particular, these cost functions aim to minimise the induced width of the junction tree which is no longer directly relevant to the performance of the algorithm. Further work may investigate other heuristics, for example with the goal of minimising cluster variance.

Finally, approximate inference algorithms have applications in learning (for example, in the gradient computations when training restricted Boltzmann machines) and thus TBP may improve learning as well as inference in some applications.

[4] TBP with $K = 10^5$ takes an average across all problem instances of approximately 10 minutes (Linkage) and 3 minutes (Promedus) per problem on our machine, and does not differ substantially for $r = 2$ vs $r = 4$. The results described by Zhu & Ermon (2015) were comparable in running time to 10 million iterations of Gibbs sampling, which we estimate would take several hours on our machine without parallelism.

Acknowledgements

We thank the anonymous reviewers for their helpful comments. This work is supported by NUS AcRF Tier 1 grant R-252-000-639-114 and a QUT Vice-Chancellor's Research Fellowship Grant.

References

Andrieu, Christophe, De Freitas, Nando, Doucet, Arnaud, and Jordan, Michael I. An introduction to MCMC for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

Bader, Brett W., Kolda, Tamara G., et al. MATLAB Tensor Toolbox Version 2.6. http://www.sandia.gov/~tgkolda/TensorToolbox/, February 2015.

Chavira, Mark and Darwiche, Adnan. Compiling Bayesian Networks with Local Structure. In *IJCAI*, 2005.

Darwiche, Adnan. A Differential Approach to Inference in Bayesian Networks. In *Uncertainty in Artificial Intelligence (UAI)*, 2000.

Gogate, Vibhav. UAI 2014 Inference Competition. http://www.hlt.utdallas.edu/~vgogate/uai14-competition/index.html, 2014.

Gogate, Vibhav and Domingos, Pedro. Structured Message Passing. In *Uncertainty in Artificial Intelligence (UAI)*, 2013.

Guennebaud, Gaël, Jacob, Benoît, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

Ihler, Alexander T and McAllester, David A. Particle Belief Propagation. In *AISTATS*, pp. 256–263, 2009.

Jensen, Finn Verner, Olesen, Kristian G, and Andersen, Stig Kjaer. An algebra of Bayesian belief universes for knowledge-based systems. *Networks*, 20(5):637–659, 1990.

Kolda, Tamara G and Bader, Brett W. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

Koller, D. and Friedman, N. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive Computation and Machine Learning. MIT Press, 2009.

Koller, Daphne, Lerner, Uri, and Angelov, Dragomir. A general algorithm for approximate inference and its application to hybrid Bayes nets. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 324–333. Morgan Kaufmann Publishers Inc., 1999.

Lauritzen, Steffen L and Spiegelhalter, David J. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 157–224, 1988.

Lee, Daniel D and Seung, H Sebastian. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 556–562, 2001.

McAuley, Julian J. and Caetano, Tibério S. Faster Algorithms for Max-Product Message-Passing. *Journal of Machine Learning Research*, 12:1349–1388, 2011.

Minka, Thomas P. Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth conference on Uncertainty in Artificial Intelligence*, pp. 362–369. Morgan Kaufmann Publishers Inc., 2001.

Mooij, Joris M. libDAI: A Free and Open Source C++ Library for Discrete Approximate Inference in Graphical Models. *Journal of Machine Learning Research*, 11: 2169–2173, August 2010.

Park, James D. and Darwiche, Adnan. A Differential Semantics for Jointree Algorithms. In *Advances in Neural Information Processing Systems (NIPS)*, 2002.

Pearl, Judea. Reverend Bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pp. 133–136, 1982.

Shafer, Glenn R and Shenoy, Prakash P. Probability propagation. *Annals of Mathematics and Artificial Intelligence*, 2(1-4):327–351, 1990.

Sudderth, Erik B, Ihler, Alexander T, Isard, Michael, Freeman, William T, and Willsky, Alan S. Nonparametric Belief Propagation. *Communications of the ACM*, 53 (10):95–103, 2010.

Wainwright, Martin J and Jordan, Michael I. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Xue, Yexiang, Ermon, Stefano, Lebras, Ronan, Gomes, Carla P, and Selman, Bart. Variable Elimination in Fourier Domain. In *Proc. 33rd International Conference on Machine Learning*, 2016.

Zhu, Michael and Ermon, Stefano. A Hybrid Approach for Probabilistic Inference using Random Projections. In *ICML*, pp. 2039–2047, 2015.