On Improving the Efficiency of the Iterative Proportional Fitting Procedure

Yee Whye Teh and Max Welling
Department of Computer Science
University of Toronto
10 King's College Road
Toronto M5S 3G4 Canada
{ywteh,welling}@cs.toronto.edu

Abstract

Iterative proportional fitting (IPF) on junction trees is an important tool for learning in graphical models. We identify the propagation and IPF updates on the junction tree as fixed point equations of a single constrained entropy maximization problem. This allows a more efficient message updating protocol than the well known effective IPF of Jiroušek and Přeučil (1995). When the junction tree has an intractably large maximum clique size we propose to maximize an approximate constrained entropy based on region graphs (Yedidia et al., 2002). To maximize the new objective we propose a "loopy" version of IPF. We show that this yields accurate estimates of the weights of undirected graphical models in a simple experiment.

1 INTRODUCTION

Junction trees are widely used as efficient representations for probability models defined on graphs. For instance, to perform exact inference in Bayesian networks one typically transforms the directed graph into a junction tree and computes the posterior probability over the cliques of the junction tree using local propagation rules. Two out of many well known schemes for this purpose are Hugin propagation (Jensen, 1996) and Shafer-Shenoy propagation (Shafer & Shenoy, 1990).

Junction trees are also indispensable for learning graphical models from data through the *iterative proportional fitting* (IPF) procedure, otherwise known as *iterative scaling* (Jiroušek & Přeučil, 1995). This *effective IPF* procedure represents the joint probability distribution in terms of the clique marginals of the junction tree, and alternates between updating the parameters of the model using IPF and propagating that change to the rest of the model using the junction tree. For structured problems, the junction tree representation reduces the space and time complexity of the IPF procedure drastically.

The first result we present in this paper is a further decrease in the time complexity of the effective IPF procedure. It is shown that both the IPF and junction tree propagation updates are fixed point equations of a maximum entropy problem with certain constraints. This unifying view lifts the strict separation in the effective IPF procedure between IPF and junction tree propagation, and allows for more efficient schedulings of the IPF and junction tree propagation updates.

For some graphs the maximum clique size in the corresponding junction tree is still intractably large, and the problem needs to be tackled through approximations. To this extent we propose a framework closely related to an exciting recent technique for approximate inference variously known as loopy belief propagation, sum-product algorithm, or generalized distributive law (Yedidia et al., 2002). Using knowledge of the close relationship between propagation and IPF updates, we propose a procedure that performs approximate IPF on *region graphs*, which are natural extensions of junction trees that may contain cycles and be designed to have smaller clique sizes. This *loopy iterative scaling* procedure consists of running fixed point equations which solve for stationary points of a constrained approximate entropy similar to the region graph free energies.

In section 2 we describe the maximum entropy problem that is the focus of the paper, as well as the classical iterative scaling algorithm. We also show the relationship between maximum entropy and maximum likelihood learning of graphical models. In section 3 we describe the effective IPF procedure. Section 4 then describes the unifying view, as well as our efficient schedule. Section 5 deals with approximate IPF on region graphs, and section 6 shows in a simple experiment the efficacy of the approximation. Section 7 closes with some discussion and extensions.

2 MAXIMUM ENTROPY

Let V be a set of nodes. Each node $i \in V$ is associated with a variable X_i. Denote the finite domain of values of X_i by $\mathbb{X}_i$ and let $x_i \in \mathbb{X}_i$ be a value of X_i. For a set of nodes $v \subset V$ let $X_v = (X_i)_{i \in v}$ be the variable associated with the nodes in v, $\mathbb{X}_v = \prod_{i \in v} \mathbb{X}_i$ be the domain of X_v, and $x_v \in \mathbb{X}_v$ be values of X_v. For simplicity we write $X_V = X$ and $X_{\backslash v} = X_{V \backslash v}$; similarly for x and $\mathbb{X}$.

Let A be a family of subsets (clusters) of V. On each cluster $\alpha \in A$ we are given a joint distribution $\widehat{p}_\alpha(x_\alpha)$ over the random variables X_α[1]. The family of distributions $\{\widehat{p}_\alpha\}_{\alpha \in A}$ is consistent if there is a distribution $P(x)$ satisfying the marginals $P(x_\alpha) = \widehat{p}_\alpha(x_\alpha)$ for all $\alpha \in A$. In this paper we assume that $\{\widehat{p}_\alpha\}$ is indeed consistent. In such a case let the maximum entropy extension be

$$\underset{P}{\operatorname{argmax}} \left\{ H(P) \,\middle|\, P(x_\alpha) = \widehat{p}_\alpha(x_\alpha) \, \forall \alpha \in A \right\} \quad (1)$$

where the entropy is $H(P) = -\sum_x P(x) \log P(x)$ and the domain of maximization is over the probability simplex.

We use Lagrange multipliers $\lambda_\alpha(x_\alpha)$ to impose the marginal constraints and γ to enforce normalization ($\sum_x P(x) = 1$). The Lagrangian is

$$\mathcal{L} = H(P) - \sum_{\alpha, x_\alpha} \lambda_\alpha(x_\alpha) \Big(\widehat{p}_\alpha(x_\alpha) - \textstyle\sum_{x_{\backslash\alpha}} P(x) \Big) - \gamma \Big(1 - \textstyle\sum_x P(x) \Big) \quad (2)$$

Zeroing derivatives of $\mathcal{L}$ with respect to $P(x)$ and γ,

$$P(x) = e^{\sum_\alpha \lambda_\alpha(x_\alpha) + \gamma - 1} \quad (3)$$

$$\gamma = 1 - \log \sum_x e^{\sum_\alpha \lambda_\alpha(x_\alpha)} \quad (4)$$

This expresses the primal variables $P(x)$ in terms of the dual variables $\lambda_\alpha(x_\alpha)$. Finally, to solve for $\lambda_\alpha(x_\alpha)$, we substitute (3,4) into (2) to obtain the dual cost

$$\mathcal{L}' = -\sum_{\alpha, x_\alpha} \lambda_\alpha(x_\alpha) \widehat{p}_\alpha(x_\alpha) + \log \sum_x e^{\sum_\alpha \lambda_\alpha(x_\alpha)} \quad (5)$$

Because the original cost function $H(P)$ is concave, its maximum coincides with the minimum of the dual cost function, and the maximum entropy extension is given in terms of the optimal $\lambda_\alpha(x_\alpha)$. Now the dual cost function $\mathcal{L}'$ is convex and can be solved by coordinate-wise descent in $\lambda_\alpha(x_\alpha)$. This is the classical iterative scaling algorithm (Deming & Stephan, 1940), given by the following updates:

$$\lambda_\alpha(x_\alpha) \leftarrow \lambda_\alpha(x_\alpha) + \log \frac{\widehat{p}_\alpha(x_\alpha)}{P(x_\alpha)} \quad (6)$$

where $P(x)$ is given by (3,4). In terms of the primal variables $P(x)$, we can understand each update of (6) as setting the marginal $P(x_\alpha)$ to be $\widehat{p}_\alpha(x_\alpha)$. In fact, (6) is equivalent to the following primal update:

$$P(x) \leftarrow P(x) \frac{\widehat{p}_\alpha(x_\alpha)}{P(x_\alpha)} \quad (7)$$

The maximum entropy framework is intimately related to maximum likelihood learning of undirected graphical models (Della Pietra et al., 1997). Let the clusters of the graphical model be given by A. The distribution expressed by the graphical model has the form

$$P(x) = \frac{1}{Z} \exp\left(\textstyle\sum_\alpha \lambda_\alpha(x_\alpha) \right) \quad (8)$$

where $\lambda_\alpha(x_\alpha)$ are the parameters of the model and Z is the normalizing partition function. Let $\widehat{p}(x)$ be an empirical distribution obtained from a set of fully observed training data. The average log likelihood of the data is then

$$\sum_x \widehat{p}(x) \log P(x) = \sum_\alpha \widehat{p}(x_\alpha) \lambda_\alpha(x_\alpha) - \log Z \quad (9)$$

which is easily seen to be the negative of the maximum entropy dual cost function (5). Further, the distribution (8) is equivalent to (3,4), hence the form of the graphical model can be derived from maximum entropy considerations with marginal constraints. Note that in this case the given distributions $\widehat{p}_\alpha(x_\alpha)$ are simply the marginal distributions $\widehat{p}(x_\alpha)$ of the empirical distribution, hence they are always consistent.

3 JUNCTION TREES

The straight forward implementation of iterative scaling uses a simple probability table to represent $P(x)$. As $|\mathbb{X}|$ grows exponentially with $|V|$ the computational cost of the algorithm is high: each iterative scaling update requires $O(|\mathbb{X}|)$ time and $O(|\mathbb{X}|)$ space.

An improved implementation, known as effective IPF, uses a junction tree to represent $P(x)$ (Jiroušek & Přeučil, 1995). At each step of iterative scaling, our distribution P is given by (8), which has the structure of an undirected graphical model with nodes V and clusters A. The corresponding graph has an edge between two nodes if both are in the same cluster $\alpha \in A$. After triangulating this graph, the maximal cliques C form a junction tree with separators S separating

[1] The extension to being given feature expectations $\widehat{f}_\alpha = \langle f_\alpha(x) \rangle$ is straight-forward and described in section 7.

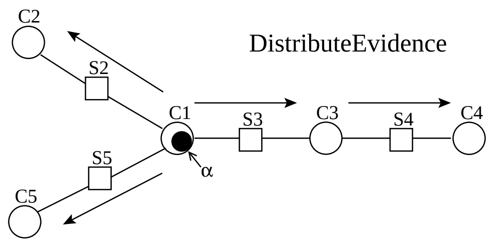

Figure 1: An ordering satisfying the running intersection property to distribute the iterative scaling change at c_1 to the rest of the graph.

them. By construction, each cluster $\alpha \in A$ is contained in some maximal clique $c \in C$, therefore $P(x)$ is decomposable with respect to the junction tree, i.e.

$$P(x) = \frac{\prod_{c \in C} P(x_c)}{\prod_{s \in S} P(x_s)} \tag{10}$$

Rather than representing $P(x)$ as a straight probability table, we represent it as a set of smaller tables $\{P_c(x_c)\}_{c \in C}$ on the cliques[2]. These tables have to be consistent, i.e. if c_1, c_2 are neighbouring cliques with s separating them, then $P_{c_1}(x_s) = P_{c_2}(x_s)$[3].

Consider the primal iterative scaling update (7). Let $c_1 \in C$ be a clique containing α. The iterative scaling update can be performed on c_1 rather than over all V:

$$P_{c_1}(x_{c_1}) \leftarrow P_{c_1}(x_{c_1}) \frac{\widehat{p}_\alpha(x_\alpha)}{P_{c_1}(x_\alpha)} \tag{11}$$

This changes the distribution $P_{c_1}(x_{c_1})$ and makes it inconsistent with the other tables. To maintain consistency, we propagate this change to the rest of the junction tree using a standard DistributeEvidence phase[4]. This is illustrated in figure 1. Let $c_1, c_2, \ldots$ be an ordering of the cliques satisfying the *running intersection property*: for each t there is a unique $\sigma(t) < t$ with $s_t \doteq c_t \cap c_{\sigma(t)} = c_t \cap (\cup_{t'<t} c_{t'})$. DistributeEvidence then amounts to

$$P_{c_t}(x_{c_t}) \leftarrow P_{c_t}(x_{c_t}) \frac{P_{c_{\sigma(t)}}(x_{s_t})}{P_{c_t}(x_{s_t})} \tag{12}$$

for $t = 2, 3, \ldots$. In essence, we are replacing the marginal $P_{c_t}(x_{s_t})$ with the new marginal $P_{c_{\sigma(t)}}(x_{s_t})$ and the information carried in the marginals flows outward from the original cluster α.

[2]The tables on the separators are not required as they can be computed by marginalizing a neighbouring clique.

[3]Because the cliques form a tree, this *local* consistency is equivalent to the *global* consistency we encountered for $\{\widehat{p}_\alpha(x_\alpha)\}$. This is unfortunately not true if the cliques do not form a tree (see section 5).

[4]Equivalently, we can use a CollectEvidence phase *before* each iterative scaling update to compute the required marginal $P_{c_1}(x_{c_1})$ (Bach & Jordan, 2002).

When the cliques are relatively small, the junction tree representation of $P(x)$ is much more efficient than a straight probability table. Let $M = \max_{c \in C} |\mathbb{X}_c| \ll |\mathbb{X}|$. Each iterative scaling update is followed by $|S|$ propagation updates. So both the time and storage requirements are $O(|S|M)$ per iterative scaling update.

4 UNIFYING PROPAGATION AND SCALING

In section 3 we introduced junction trees as simply a computational tool to improve the efficiency of the iterative scaling procedure. We will show in this section that both the propagation updates (12) and the iterative scaling updates (11) can be derived as fixed point equations of a constrained maximization problem. A consequence of this is that any intermixed schedule of iterative scaling and junction tree propagation updates will converge to the maximum entropy solution. This allows us more flexibility in designing efficient schedules of the updates. In particular, we propose a particular scheduling which requires only $2|S|$ propagation updates to perform all $|A|$ iterative scaling updates once. This is more efficient than the algorithm in section 3.

4.1 CONSTRAINED MAXIMIZATION

Consider the following constrained maximization problem:

$$\operatorname*{argmax}_{\{P_c, P_s\}} \Big\{ \sum_c H(P_c) - \sum_s H(P_s) \,\Big|\, P_c(x_\alpha) = \widehat{p}_\alpha(x_\alpha), \\ P_c(x_s) = P_s(x_s) \, \forall \alpha, s, c \text{ with } \alpha, s \subset c \Big\} \tag{13}$$

where the domains of P_c and P_s are probability simplexes. When the constraints are satisfied, the distributions on the cliques and separators are consistent hence they can be combined into a single distribution $P(x)$ using (10). Then the cost function is

$$\sum_c H(P_c) - \sum_s H(P_s) = H(P) \tag{14}$$

This means that (13) is a specific case of the original maximum entropy problem (1) where P is assumed decomposable with respect to the junction tree. But section 3 shows that the maximum entropy extension is itself decomposable with respect to the same junction tree. Hence the marginal distributions of the maximum entropy extension form a solution to (13) and the two problems are actually equivalent.

Again we will use Lagrange multipliers to solve (13). Let $\lambda_{cs}(x_s)$ impose the marginal consistencies and let γ_c, γ_s make sure P_c and P_s are normalized. We identify

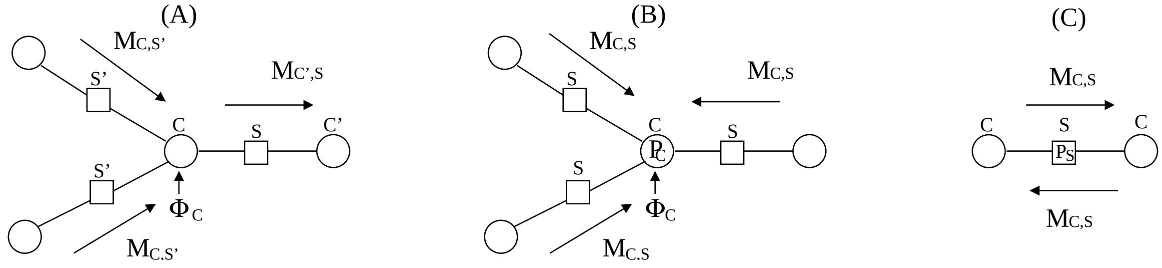

Figure 2: (A) Shafer-Shenoy propagation updates. (B) Computing clique distributions from messages. (C) Computing separator distributions from messages.

each $\alpha \in A$ with a clique $c_\alpha \in C$ and let $\lambda_\alpha(x_\alpha)$ impose the given constraint that $P_{c_\alpha}(x_\alpha) = \widehat{p}_\alpha(x_\alpha)$. Let $A_c = \{\alpha \in A | c_\alpha = c\}$. The Lagrangian is then

$$\mathcal{L} = \sum_c H(P_c) - \sum_s H(P_s) - \sum_{v \in S \cup C} \gamma_v \Big(\textstyle\sum_{x_v} P_v(x_v) - 1 \Big)$$
$$- \sum_{c,s,x_s} \lambda_{cs}(x_s) \Big(P_s(x_s) - \textstyle\sum_{x_{c \setminus s}} P_c(x_c) \Big)$$
$$- \sum_{\alpha, x_\alpha} \lambda_\alpha(x_\alpha) \Big(\widehat{p}_\alpha(x_\alpha) - \textstyle\sum_{x_{c_\alpha \setminus \alpha}} P_{c_\alpha}(x_{c_\alpha}) \Big) \quad (15)$$

Solving the Lagrangian as before, we find that the marginal distributions are

$$P_c(x_c) \propto e^{\sum_s \lambda_{cs}(x_s) + \sum_{\alpha \in A_c} \lambda_\alpha(x_\alpha)} \quad (16)$$
$$P_s(x_s) \propto e^{\sum_c \lambda_{cs}(x_s)} \quad (17)$$

while λ_α and λ_{cs} are updated with the fixed point equations

$$\lambda_\alpha(x_\alpha) \leftarrow \lambda_\alpha(x_\alpha) + \log \frac{\widehat{p}_\alpha(x_\alpha)}{P_{c_\alpha}(x_\alpha)} \quad (18)$$
$$e^{\lambda_{c's}(x_s)} \leftarrow\propto \sum_{x_{c \setminus s}} e^{\sum_{s' \neq s} \lambda_{cs'}(x_{s'}) + \sum_\alpha \lambda_{c\alpha}(x_\alpha)} \quad (19)$$

where c' and c are the two cliques separated by s, and s' are other separators neighbouring c. We see that iterative scaling updates (6) are fixed point equations (18) to solve the Lagrangian. Also identifying messages and potentials as

$$M_{cs}(x_s) \doteq e^{\lambda_{cs}(x_s)} \qquad \phi_c(x_c) \doteq e^{\sum_\alpha \lambda_{c\alpha}(x_\alpha)} \quad (20)$$

(19) is easily shown to be

$$M_{c's}(x_s) \leftarrow\propto \sum_{x_{c \setminus s}} \phi_c(x_c) \prod_{s' \neq s} M_{cs'}(x_{s'}) \quad (21)$$

which can be identified as a Shafer-Shenoy propagation update for junction trees (Shafer & Shenoy, 1990). The marginal distributions are then given by

$$P_c(x_c) \propto \phi_c(x_c) \prod_s M_{cs}(x_s) \quad (22)$$

$$P_s(x_s) \propto \prod_c M_{cs}(x_s) \quad (23)$$

The Shafer-Shenoy updates are depicted in figure 2.

Sometimes, it is more intuitive and effective to perform iterative scaling using the primal updates of (7) rather than the dual updates of (6). Similarly, sometimes propagation updates which deal directly with clique marginals are more desirable. One of these is Hugin propagation, given by

$$P_{c'}(x_{c'}) \leftarrow P_{c'}(x_{c'}) \frac{P_c(x_s)}{P_s(x_s)} \quad P_s(x_s) \leftarrow P_c(x_s) \quad (24)$$

which can be shown to be equivalent to Shafer-Shenoy.

4.2 EFFICIENT SCHEDULING

The previous subsection shows that both iterative scaling and Shafer-Shenoy propagation updates are fixed point equations to solve the maximum entropy problem (13). Because the cost function of (13) is concave in the space where the constraints are satisfied, the fixed point equations are guaranteed to converge to the global optimum. However this does not imply anything about the efficiency of various schedules. We now propose a particular class of schedules which will be efficient in the sense to be defined below.

We can understand the iterative scaling update (18) as changing the Lagrange multiplier $\lambda_{c\alpha}(x_\alpha)$, given the current estimate $P_c(x_c)$ of the true marginal distribution $P(x_c)$, so as to satisfy the constraint $P(x_\alpha) = \widehat{p}_\alpha(x_\alpha)$. On the other hand, the propagation updates (19) or (21) compute the required marginal distributions $P(x_c)$ and store them in $P_c(x_c)$ from the current Lagrange multipliers. If the propagation updates are run until convergence after every iterative scaling update, then the given $P_c(x_c)$ will be the true $P(x_c)$. This is the schedule of the effective IPF procedure. However it is clear that this schedule is inefficient since it calculates all the marginal distributions exactly even though only one is needed for the next iterative scaling update. On the other hand, if the propagation updates have not converged before an iterative scaling update is performed, then the calculated marginal distribution $P_c(x_c)$ might not be exact. As a result the

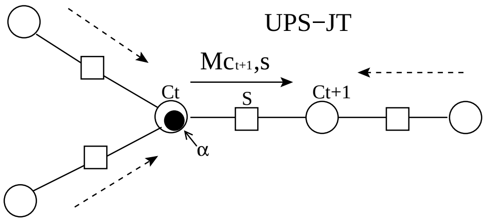

Figure 3: The dashed lines are the messages which are still correct, while the solid line denotes the message that is updated to become correct.

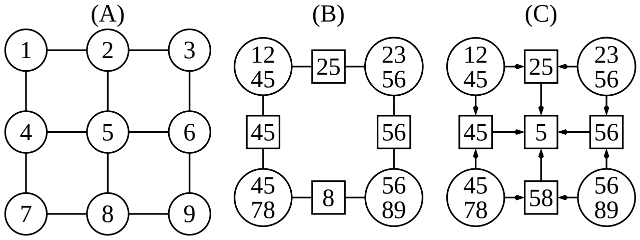

Figure 4: (A) An example of a graphical model. (B) A junction graph for the model. (C) A region graph constructed using the cluster variational method.

iterative scaling update might not be as effective.

In view of the above issues, we shall show that our proposed schedule, unified propagation and scaling for junction trees (UPS-JT), is efficient in that it satisfies the following properties: (1) whenever an iterative scaling update is performed, the current estimate of the required marginal $P_c(x_c)$ is exact; (2) between any two iterative scaling updates, only at most one propagation update is performed to ensure $P_c(x_c)$ is exact for the second iterative scaling update.

Unified Propagation and Scaling for Junction Trees

---

1. Initialize the junction tree so that each P_c and P_s is uniform.
2. Initialize messages or Lagrange multipliers to uniform as well.
3. Initialize c_1 to some clique in C.
4. For $t = 1, 2, \ldots$ until convergence criterion is met:
5.   Perform iterative scaling updates for those clusters $\alpha \in A_{c_t}$ identified with c_t.
6.   Choose a clique c_{t+1} neighbouring c_t.
7.   Perform the propagation update from c_t to c_{t+1}.

Note that UPS-JT does not prescribe the ordering in which we visit the cliques except for the implicit requirement that all cliques are visited enough times. One possible ordering is to visit the cliques in a depth first search manner on the junction tree. This guarantees that every Lagrange multiplier is updated once for a total of $2|S|$ propagation updates. This is much more efficient than the ordering in the effective IPF procedure (Jiroušek & Přeučil, 1995), which takes $|S|$ propagation updates for every iterative scaling update.

Compared with the effective IPF procedure, we see that UPS-JT performs multiple iterative scaling updates on each clique, and, more importantly, substituted a full DistributeEvidence phase with a single propagation update. Hence it satisfies condition 2 above. We now prove by induction that condition 1 is satisfied when using Shafer-Shenoy propagation. The inductive hypothesis is that at each iteration all incoming messages to c_t are correct. At time $t = 1$ this is trivially true. In figure 3 we depict one step of the UPS-JT algorithm for time $t \geq 1$. we have updated the Lagrange multipliers $\lambda_\alpha(x_\alpha)$ in clique c_t. Next, as required by step 7 of the UPS-JT algorithm, we choose a neighboring clique in the tree, c_{t+1}, to perform our next scaling update. As required by condition 1, we need the exact marginal $P(x_{c_{t+1}})$. To compute that we collect evidence to the clique c_{t+1}, by propagating inward from the leaves. But note that all messages (e.g. dashed arrows in figure 3), except message $M_{c_{t+1}s}(x_s)$ (solid arrow), are unchanged by the scaling update at node c_t. Hence, we only recompute the latter and use (16) to get the correct marginal at clique c_{t+1}.

5 LOOPY ITERATIVE SCALING

UPS-JT is an efficient algorithm for entropy maximization if the cliques of the junction tree are small. However, its complexity scales exponentially with the size of the maximal clique. To combat this, we propose an approximate algorithm named *loopy iterative scaling* based on *region graphs*.

5.1 REGION GRAPHS

After Yedidia et al. (2002) we define a region graph as an acyclic directed graph where the vertices are labelled with subsets of nodes, or regions. The top layer consists of a family of large regions which cover the original graph, such that each cluster $\alpha \in A$ is contained in at least one of them. A directed edge can only exist between a parent and a child if the nodes associated with the child are a subset of the nodes associated with its parent. With each vertex (or region) we will also associate a "counting number" c_r,

$$c_r = 1 - \sum_{r' \in \mathrm{Super}(r)} c_{r'} \tag{25}$$

where $\mathrm{Super}(r)$ consists of all regions which strictly contain r, and $c_r = 1$ for the top layer regions. A valid region graph should fulfill the following two conditions: (1) for each node, the subgraph induced by the vertices containing that node must be connected; (2) the sum of the counting numbers of each such subgraph must add up to one.

With each region graph we can associate an approximate entropy

$$\mathcal{H}(\{P_r\}) = \sum_r c_r \; H(P_r) \approx H(P). \qquad (26)$$

and generalized belief propagation algorithms to maximize it (see Yedidia et al. (2002) for details). The collection $\{P_r\}$ are now *approximate* marginals satisfying *local* consistency constraints[5]: $P_r(x_s) = P_s(x_s)$ for every child s of region r. The region-based entropy maximization problem is then given by

$$\operatorname*{argmax}_{\{P_r\}} \Big\{ \mathcal{H}(\{P_r\}) \,\Big|\, P_r(x_\alpha) = \widehat{p}_\alpha(x_\alpha), \\ P_r(x_s) = P_s(x_s) \, \forall \alpha, s, r \text{ with } \alpha, s \subset r \Big\} \qquad (27)$$

The solutions to (27) are approximations to the marginals of the true maximum entropy distribution. Further, the optimal Lagrange multipliers are approximate solutions to the maximum likelihood parameters, if we are doing maximum likelihood training of a graphical model.

Solving the above region-based entropy maximization problem, it is not hard to show that the fixed point equations are given again by the scaling updates (18) for the Lagrange multipliers, and the fixed point equations of generalized belief propagation. Here the Lagrange multipliers associated with the constraints $P_{r_\alpha}(x_\alpha) = \widehat{p}_\alpha(x_\alpha)$ are $\lambda_{r_\alpha}(x_\alpha)$, and the potentials for the top layer regions r are $\phi_r(x_r) \doteq e^{\sum_{\alpha \in A_r} \lambda_{r\alpha}(x_\alpha)}$, where r_α is some top layer region containing α, and $A_r = \{\alpha | r_\alpha = r\}$. These *loopy iterative scaling* updates can be performed using any convenient scheduling, but convergence is unfortunately not guaranteed.

Although region graphs conveniently translate into loopy iterative scaling algorithms, it has not been made clear how to construct a valid region graph given a family of large regions. There are two distinct methods described in the literature, one based on junction graphs, the other called the cluster variation method.

5.2 JUNCTION GRAPH METHOD

A junction graph is a two layer region graph with large regions called cliques and their children, called separators. Since the region graph has such a simple structure, we typically ignore the directionality of the edges. For each node we require that the subgraph constructed from all cliques and separators containing that node should form an undirected tree. Note that this condition is stronger than the region graph condition, and automatically ensures that all counting numbers in that subgraph sum to one. This property is the equivalent of the running intersection property for junction trees and guarantees that junction graphs "look like" junction trees locally. An example of a junction graph is shown in figure 4.

Given a cluster set A, there is a variety of junction graphs that are consistent with A. On one end of the spectrum, there are junction trees, on which we can perform exact entropy maximization. On the other end, we have junction graphs with small cliques that are poor approximations but admit efficient algorithms to maximize the approximate entropy. In fact, Aji and McEliece (2001) show that for any collection of subsets of nodes (in particular, for A) it is easy to construct a junction graph whose cliques consist precisely of the subsets in the collection: first define the separators as the intersections of pairs of cliques, then for every node construct the subgraph induced by the cliques and separators containing that node and delete nodes from separators as well as remove separators which are empty until the subgraph is a tree.

Junction graphs are particularly convenient because the propagation updates reduce precisely to the junction tree updates (19). Moreover, the approximate entropy (26) reduces to the following *Bethe entropy*,

$$\mathcal{H}(\{P_c, P_s\}) = \sum_c H(P_c) - \sum_s H(P_s) \qquad (28)$$

5.3 CLUSTER VARIATION METHOD

An alternative road to constructing valid region graphs is provided by the cluster variation method. We start again with a family of large regions such that each cluster is contained in at least one of them. Next, the children of the large regions are defined as their intersections[6], and the children of those are given by the intersections of the intersections. This process is repeated until no further (non-trivial) region can be added. The resultant layered region graph, with counting numbers assigned to regions using (25) is automatically valid, and the corresponding approximate entropy is known as the Kikuchi entropy. The corresponding loopy iterative scaling algorithm is now based on the generalized belief propagation algorithms as described in Yedidia et al. (2002).

[5]Note that since the region graph can contain cycles there might not be a distribution which is consistent with all $\{P_r\}$, i.e. they may not be *globally* consistent.

[6]For each layer we do not include regions which are subsets of other regions in that layer.

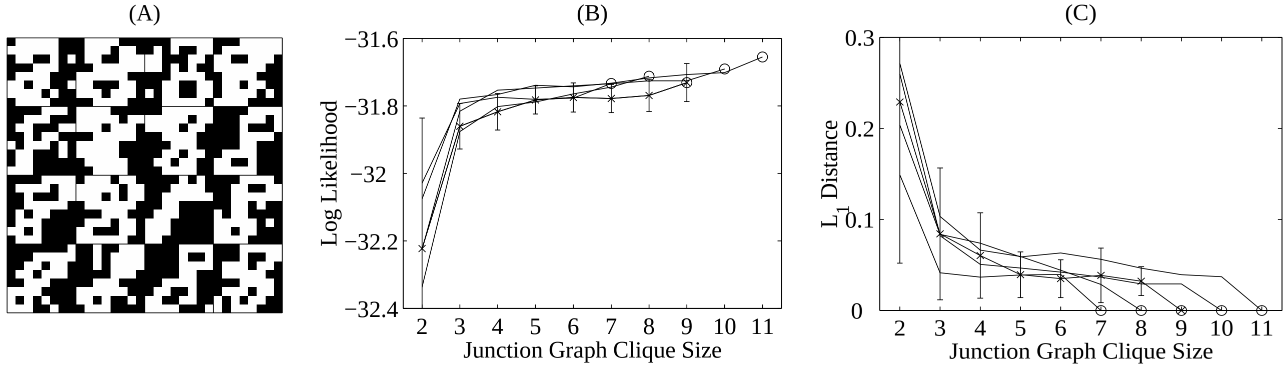

Figure 5: (A) Some examples of the images of '8's. (B) Average log likelihoods over training data as a function of junction *graph* clique size. Each curve is averaged over all models with a certain maximal clique size of the corresponding junction *tree*. For models of treewidth 9, we also plotted the standard deviation of the log likelihoods. (C) Average L_1 distances of the various models.

6 EXPERIMENT

We explored the behaviour of loopy iterative scaling on junction graphs on a simple task of learning the weights of pairwise Markov networks. The training data consists of 8×8 binary images of hand-written '8's preprocessed from the CEDAR dataset (see figure 5A).

Thirty models are generated by randomly sampling 5% of the edges connecting nodes a distance of at most 6 apart on the maximum likelihood tree. Junction trees are constructed by triangulating the resulting graphs using node elimination, where at each stage the node with the least neighbors is chosen. The maximal clique sizes of these junction trees vary between 7 and 11. For each model, starting with the edges as the cliques of the junction graph, a range of junction graphs is constructed by growing its cliques one node at a time, and making sure that the resulting cliques are contained within the cliques of the corresponding junction tree. At each stage a node is added to a clique such that the increase in the total mutual information[7] is largest, and removing cliques which are subsumed by other cliques. On all the junction graphs so constructed we used loopy iterative scaling to learn the parameters, where both scaling and propagation updates were damped in the log-domain. Propagation updates were iterated until convergence before the scaling updates were performed once.

After loopy iterative scaling has converged, we assessed the accuracy of the results using two measures. In figure 5B we plot the log likelihoods of the data under the learned models, while in figure 5C we show the L_1 distances between the empirical marginal distributions $\widehat{p}_\alpha(x_\alpha)$ and those of the learned models $P(x_\alpha)$, averaged over the edges α. The approximation is reasonably accurate, and its accuracy improves as we increase the maximum clique size of the junction graph.

[7]Total mutual information is defined as the sum of the mutual informations between all pairs of nodes within the same cliques.

7 DISCUSSION

In this paper we have shown that propagation and iterative scaling on junction trees can be unified as fixed point equations for solving a certain constrained maximum entropy problem. From this insight we have proposed a more efficient scheduling for iterative scaling on junction trees. For graphs with a maximal clique size which is prohibitively large, we have proposed a loopy iterative scaling algorithm, based on region graphs.

There are a number of important extensions of the methods discussed in this paper. Firstly, rather than finding a distribution with maximum entropy we can find a distribution with minimum relative entropy (KL divergence) to a given distribution P_0. The relative entropy from P to P_0 is defined to be

$$D(P|P_0) = \sum_x P(x) \log \frac{P(x)}{P_0(x)} \tag{29}$$

When P_0 is the uniform distribution, minimizing $D(P|P_0)$ is equivalent to maximizing the entropy of P. This extension is useful when we would like to learn undirected graphical models where certain potentials are fixed. These potentials are cast into P_0 in the minimum relative entropy framework. Our results on unifying junction tree propagation and iterative scaling carry through if P_0 is decomposable with respect to the junction tree as well.

Secondly, the results are straightforwardly extended to constraints on feature expectations, rather than on marginals. Here each cluster $\alpha \in A$ is associated with a vector-valued feature $f_\alpha : \mathbb{X}_\alpha \to \mathbb{R}^{d_\alpha}$, and $P(x)$ is constrained to have $\sum_x P(x) f_\alpha(x_\alpha) = \widehat{f}_\alpha \in \mathbb{R}^{d_\alpha}$. When dualized, the maximum entropy problem becomes a maximum likelihood problem for an undirected graphical model where f_α are the features. The Lagrange multipliers $\lambda_\alpha \in \mathbb{R}^{d_\alpha}$ imposing the expectation constraints become the weights corresponding to the features in the undirected graphical model

$$P(x) = \frac{1}{Z} e^{\sum_\alpha \lambda_\alpha^T f_\alpha(x_\alpha)} \tag{30}$$

The required expectations $\widehat{f}_\alpha$ are obtained by averaging over a training set. The algorithms discussed in this paper therefore open the way for more efficient training of maximum entropy models (Della Pietra et al., 1997), conditional maximum entropy models (Lafferty et al., 2001) and thin junction trees (Bach & Jordan, 2002).

There is an interesting link between minimum divergence problems and inference. If the marginal constraints put all the probability mass on a single state, i.e. $\widehat{p}_i(x_i) = \delta_{x_i, \widehat{x}_i}$ for some $\widehat{x}_i$, then the two problems become equivalent (Teh & Welling, 2002). This implies that the generalized distributive law of Aji and McEliece (2001) and the generalized belief propagation algorithms are in fact special cases of loopy iterative scaling on junction graphs and region graphs respectively. We are currently studying the possibility of extending the convergent version of UPS described in (Teh & Welling, 2002) to the more general approximations discussed in this paper.

In this paper we have proposed novel algorithms for learning *fully observed* undirected graphical models. When the models are *partially observed*, a standard method to train them is the EM algorithm. When the posterior distribution is intractable, a number of researchers have looked at approximating the E steps with loopy belief propagation (Frey & Kannan, 2001). Because there is no global cost function which both the E and M steps are minimizing, we cannot make any statements on the accuracy or convergence properties of such algorithms. An exciting research direction is to extend our framework to the partially observed case, where we now have an approximate EM algorithm where both E and M steps are derived as fixed point equations minimizing a region-based free energy.

Acknowledgments

We thank Francis Bach and Michael Jordan for interesting discussions and Geoffrey Hinton for inspiration and support.

References

Aji, S. M., & McEliece, R. J. (2001). The generalized distributive law and free energy minimization. *Proceedings of the Allerton Conference on Communication, Control, and Computing.*

Bach, F. R., & Jordan, M. I. (2002). Thin junction trees. *Advances in Neural Information Processing Systems.*

Della Pietra, S., Della Pietra, V., & Lafferty, J. (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19.*

Deming, W. E., & Stephan, F. F. (1940). On a least square adjustment of a samplied frequency table when the expected marginal totals are known. *Annals of Mathematical Statistics*, *11*, 427–444.

Frey, B., & Kannan, A. (2001). Accumulator networks: suitors of local probability propagation. *Advances in Neural Information Processing Systems.*

Jensen, F. V. (1996). *An introduction to bayesian networks*. UCL Press, London.

Jiroušek, R., & Přeučil, S. (1995). On the effective implementation of the iterative proportional fitting procedure. *Computational Statistics And Data Analysis*, *19*, 177–189.

Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Propabilistic models for segmenting and labeling sequence data. *Proceedings of the International Conference on Machine Learning.*

Shafer, G. R., & Shenoy, P. P. (1990). Probability propagation. *Annals of Mathematics and Artificial Intelligence*, *2*, 327–352.

Teh, Y. W., & Welling, M. (2002). The unified propagation and scaling algorithm. *Advances in Neural Information Processing Systems.*

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2002). *Constructing free energy approximations and generalized belief propagation algorithms* (Technical Report TR-2002-35). Mitsubishi Electric Research Laboratories.