Efficient Inference with Cardinality-based Clique Potentials

Rahul Gupta RAHULGUPTA@IN.IBM.COM
IBM Research Lab, New Delhi, India

Ajit A. Diwan AAD@CSE.IITB.AC.IN
Sunita Sarawagi SUNITA@IITB.AC.IN
IIT Bombay, India

Abstract

Many collective labeling tasks require inference on graphical models where the clique potentials depend only on the number of nodes that get a particular label. We design efficient inference algorithms for various families of such potentials. Our algorithms are exact for arbitrary cardinality-based clique potentials on binary labels and for max-like and majority-like clique potentials on multiple labels. Moving towards more complex potentials, we show that inference becomes NP-hard even on cliques with homogeneous Potts potentials. We present a $\frac{13}{15}$-approximation algorithm with runtime sub-quadratic in the clique size. In contrast, the best known previous guarantee for graphs with Potts potentials is only 0.5. We perform empirical comparisons on real and synthetic data, and show that our proposed methods are an order of magnitude faster than the well-known Tree-based re-parameterization (TRW) and graph-cut algorithms.

1. Introduction

We present fast inference algorithms for graphical models where the clique potentials comprise of two parts: the first part is sum of individual node potentials and, the second part is a symmetric n-ary function that depends only on the number of nodes that get a particular label. We call these *cardinality-based* clique potentials.

Such cliques appear in graphical modeling of many applications, including collective labeling for information extraction (Sutton & McCallum, 2004; Bunescu & Mooney, 2004; Krishnan & Manning, 2006; Finkel et al., 2005), hypertext classification (Lu & Getoor, 2003), relational learning (Jensen et al., 2004), and associative networks (Taskar et al., 2004). Typically, these applications give rise to graphs with large cliques where exact inference with junction trees would require time exponential in the size of the largest clique. Existing methods have relied on approximate inference algorithms like belief propagation (Sutton & McCallum, 2004; Bunescu & Mooney, 2004; Taskar et al., 2004), sampling (Finkel et al., 2005), relaxation labeling (Lu & Getoor, 2003), or stacking (Krishnan & Manning, 2006) to address the challenge of large clique sizes.

We exploit the special form of the clique potentials to design efficient combinatorial algorithms for computing max-messages on the nodes of such cliques. This allows efficient belief propagation on cluster graphs with very large embedded cliques using the techniques of Duchi et al. (2007). One source of cardinality-based potentials is cliques with homogeneous edge potentials as shown in Section 2. Instead of viewing them as individual edge potentials, we treat them as cardinality-based clique potentials that are amenable to our special inference algorithms. These are an order of magnitude faster than state-of-the-art edge-based propagation algorithms like TRW (Kolmogorov, 2004) and can scale to cliques over thousands of nodes.

We identify various families of cardinality-based clique potentials that arise in practice. We present a $O(nm \log n)$ inference algorithm that finds exact MAP on max-like clique potentials over m-ary label space, and arbitrary cardinal potentials when $m = 2$. Next, we show that inference is NP-hard for cardinal cliques arising out of a homogeneous Potts model. In this case, we show that a $\frac{13}{15}$-approximation is possible with the above algorithm and that it can be generalized to provide an approximation ratio of $\frac{4p}{1+4p}$, that can be made arbitrarily close to 1, in $O(nm^p \log n)$ time. This algorithm provides provably better guarantees than the popular α expansion algorithm (Boykov et al., 2001) which has an approximation guarantee of $\frac{1}{2}$ on general graphs. We show that even for homogeneous Potts

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007.

models, α-expansion cannot give a bound better than $\frac{1}{2}$. Finally, for majority-like potentials we present an LP-based exact algorithm and a faster $O(nm \log n)$ approximate algorithm.

2. Applications of Cardinality-based Potentials

In this section we illustrate with examples from two applications the types of cardinality-based clique potentials that arise in practice.

2.1. Information Extraction

Classical information extraction (IE) models are based on sequential graphical models with each word a node and edges joining adjacent words indicating that their entity labels directly influence each other. Recently, several researchers (Sutton & McCallum, 2004; Bunescu & Mooney, 2004; Finkel et al., 2005; Krishnan & Manning, 2006) have reported increased accuracy of collectively labeling repeated words within a document or across multiple documents. In the corresponding graphical model this leads to additional edges between non-adjacent positions that share a word. Let the potential function for an edge between adjacent word positions $j-1$ and j in document i be $\phi_{ij}(y, y')$ and for non-adjacent positions that share a word w be $f_w(y, y')$. The goal during inference is to find a labeling $\mathbf{y}$ where y_{ij} is the label of word x_{ij} in position j of doc i, so as to maximize:

$$\sum_{i,j} \phi_{ij}(y_{ij}, y_{i(j-1)}) + \sum_{w} \sum_{\mathrm{x}_{ij}=x_{i'j'}=w} f_w(y_{ij}, y_{i'j'}) \quad (1)$$

The above inference problem gets intractable very soon with the addition of non-adjacent edges beyond the highly tractable collection of chain models of classical IE. Consequently, all prior work on collective extraction for IE relied on generic approximation techniques including belief propagation (Sutton & McCallum, 2004; Bunescu & Mooney, 2004) , Gibbs sampling (Finkel et al., 2005) or stacking (Krishnan & Manning, 2006).

We present a different view of the above inference problem using cardinality-based clique potential functions $\mathcal{C}_w()$ defined over label subsets $\mathbf{y}^w$ of positions where word w occurs. We rewrite the second term in Equation 1 as

$$\frac{1}{2} \sum_{w} \Big(\sum_{y,y'} f_w(y, y') n_y(\mathbf{y}^w) n_{y'}(\mathbf{y}^w) - \sum_{y} n_y(\mathbf{y}^w) f_w(y, y) \Big) = \sum_{w} \mathcal{C}_w(n_1(\mathbf{y}^w), \ldots n_m(\mathbf{y}^w))$$

where $n_y(\mathbf{y}^w)$ is the number of times w is labeled y in all its occurrences. The clique potential $\mathcal{C}_w$ only depends on the counts of how many nodes get assigned a particular label. A useful special case of the function is when $f_w(y, y')$ is positive only for the case that $y = y'$, and zero otherwise.

2.2. Hypertext Classification

In hypertext classification, the goal is to classify a document based on features derived from its content and labels of documents it points to. A common technique in statistical relational learning to capture the dependency between a node and the variable number of neighbors it might be related to, is to define fixed length feature vectors out of the neighbor's labels. In text classification, most previous approaches (Taskar et al., 2002; Lu & Getoor, 2003; Chakrabarti et al., 1998) have created features based on the counts of labels in its neighborhood. Accordingly, we can define the following set of potentials: a node-level potential $\phi_i(y)$ that depends on the content of the document i, and a neighborhood potential $f(y, n_1(\mathbf{y}^{O_i}), \ldots, n_m(\mathbf{y}^{O_i}))$ that captures the dependency of the label of i on the counts in the label vector $\mathbf{y}^{O_i}$ of its outlinks.

$$\begin{aligned} & \sum_{i} (\phi_{iy_i} + f(y_i, n_1(\mathbf{y}^{O_i}), \ldots, n_m(\mathbf{y}^{O_i}))) \\ = & \sum_{i} (\phi_{iy_i} + \sum_{y} \mathcal{C}_y(n_1(\mathbf{y}^{O_i}), \ldots, n_m(\mathbf{y}^{O_i})) [\![y = y_i]\!]) \end{aligned}$$

Lu and Getoor (2003) include several examples of such clique potentials, viz. the Majority potential $\mathcal{C}_y(n_1, \ldots n_m) = \phi(y, y_{max})$ where $y_{max} = \mathrm{argmax}_y n_y$, and the Count potential $\mathcal{C}_y(n_1, \ldots n_m) = \sum_{y':n_{y'}>0} \phi(y', y) n_{y'}$. Some of these potentials, for example, the Majority potential are not decomposable as sum of potentials over the edges of the clique. This implies that methods such as TRW and graph-cuts are not applicable. Lu and Getoor (2003) rely on the Iterated Conditional Modes (ICM) method that greedily selects the best label of each document in turn based on the label counts of its neighbors. We show empirically that ICM provides lower MAP scores than our algorithms for most of the potentials.

We now focus on algorithms for MAP computation over a single clique, which can then be used to compute max-marginals over its nodes. We defer the issue of efficient incremental computation of max-marginals to future work.

3. Inference Algorithms

Our goal is to find a labeling $\mathbf{y} = y_1 \ldots y_n$ of the n nodes of the clique so as to maximize the sum of their node potentials $\mathrm{np}(\mathbf{y}) = \sum_i \phi_{iy_i}$ and clique potential $\mathrm{cp}(\mathbf{y}) = \mathcal{C}(n_1(\mathbf{y}), \ldots, n_m(\mathbf{y}))$. We use $F(\mathbf{y}) =$

$\text{np}(\mathbf{y}) + \text{cp}(\mathbf{y})$ to denote the total score of labeling $\mathbf{y}$ and, $\mathbf{y}^*$ to denote the labeling with the highest score or the MAP labeling. We use $\mathbf{n}(\mathbf{y}) = n_1(\mathbf{y}), \ldots, n_m(\mathbf{y})$, where $\sum_i n_i(\mathbf{y}) = n$, to denote a vector of counts. Also, to avoid the problem of arbitrary scaling of parameters, we assume wlog that $\phi_{um} = 0$ for all nodes.

When the number of labels $m = 2$, MAP can be found exactly in $O(n \log n)$ time for arbitrary clique potentials using the following simple strategy. Sort the nodes in decreasing order of their $\phi_{u1} - \phi_{u2}$ values. For each number k from 0 to n, find s_k the sum of the top-k node potentials $\phi_{u1} - \phi_{u2}$ and clique potential $\mathcal{C}(k, n-k)$. The largest s_k gives the optimal labeling.

This result also implies that in the binary case when the graph is a clique and all edges have the same potential, then for arbitrary such potentials we can find the optimal labeling in $O(n \log n)$ time. In contrast, for general graphs in the binary case, exact algorithms are known only for associative edge potentials (Boykov et al., 2001; Kolmogorov & Wainwright, 2005)

When the number of labels $m > 2$, finding the MAP with arbitrary clique potentials is hard. A generalization of the above algorithm to the multi-label case is to consider all possible partitions of the total count n and for a fixed partition $n_1, \ldots, n_m$ assign n_y nodes label y. The optimal assignment can be found using maximum matching on the node potentials. The worst case complexity of this algorithm is $O(n^m)$, so it is not too useful except for very small values of m. We show that the problem is NP-hard even for some very simple potential functions.

We consider specific families of clique potentials, many of which are currently popular, and design various exact and approximate algorithms that exploit the specific structure of the potential. In particular, we consider the following three types of clique potentials:

1. MAX: $\mathcal{C}(n_1, \ldots, n_m) = \max_y f_y(n_y)$. These have been used for analyzing associative networks in (Taskar et al., 2004).

2. SUM: $\mathcal{C}(n_1, \ldots, n_m) = \sum_y f_y(n_y)$. A common example of this function is $\lambda \sum_y n_y^2$ that arises out of cliques with all edges following the same Potts potential with parameter λ. Another important example of this class is negative entropy where $f_y(n_y) = n_y \log n_y$.

3. MAJORITY: $\mathcal{C}(n_1, \ldots, n_m) = f_a(\mathbf{n})$, where $a = \text{argmax}_y n_y$. One common example of such a potential, as used in (Krishnan & Manning, 2006) and (Lu & Getoor, 2003), is $f_a(\mathbf{n}) = \sum_y w_{ay} n_y$, where $\{w_{yy'}\}$ is an arbitrary matrix.

3.1. MAX Clique Potentials

When the clique potentials are of the form $\mathcal{C}(n_1 \ldots n_m) = \max_y f_y(n_y)$ for arbitrary functions f_y, we can find the MAP using the following algorithm.

THE α-PASS ALGORITHM

For each label α between 1 and m, sort the nodes in decreasing order of $\phi_{u\alpha} - \max_{y \neq \alpha} \phi_{uy}$ where ϕ_{uy} denotes the node potential of label y on u. In a single sweep over the sorted nodes, find for each k, the labeling $\hat{\mathbf{y}}^{\alpha k}$ obtained by assigning the first k nodes label α and the remaining nodes their best label other than α. Choose the best labeling $\hat{\mathbf{y}} = \text{argmax} F(\hat{\mathbf{y}}^{\alpha k})$

The algorithm runs in $O(nm \log n)$ time by incrementally computing $\max F(\hat{\mathbf{y}}^{\alpha k})$ from $\max F(\hat{\mathbf{y}}^{\alpha(k-1)})$.

Claim 3.1. *Labeling $\hat{\mathbf{y}}^{\alpha k}$ has the maximum node score over all $\mathbf{y}$ where k nodes are labeled α, that is, $np(\hat{\mathbf{y}}^{\alpha k}) = max_{\mathbf{y}:n_\alpha(\mathbf{y})=k} np(\mathbf{y})$.*

Claim 3.2. *For MAX potentials, $\mathcal{C}(\hat{\mathbf{y}}^{\alpha k}) \geq f_\alpha(k)$.*

Theorem 3.1. *The α-pass algorithm finds the MAP for MAX clique potentials.*

Proof. Let $\mathbf{y}^*$ be the optimal labeling and let $\beta = \text{argmax}_j f_j(n_j(\mathbf{y}^*)), \ell = n_\beta(\mathbf{y}^*)$. Let $\hat{\mathbf{y}}$ be the labeling found by α-pass. $F(\hat{\mathbf{y}}) = \max_{1 \leq \alpha \leq m, 1 \leq k \leq n} F(\hat{\mathbf{y}}^{\alpha k}) \geq F(\hat{\mathbf{y}}^{\beta \ell}) = \text{np}(\hat{\mathbf{y}}^{\beta \ell}) + \text{cp}(\hat{\mathbf{y}}^{\beta \ell}) \geq \text{np}(\hat{\mathbf{y}}^{\beta \ell}) + f_\beta(\ell) = \text{np}(\hat{\mathbf{y}}^{\beta \ell}) + \text{cp}(\mathbf{y}^*) \geq \text{np}(\mathbf{y}^*) + \text{cp}(\mathbf{y}^*)$ The last inequality follows from Claim 3.1. □

3.2. SUM Clique Potentials

SUM clique potentials are of the form $\mathcal{C}(n_1, \ldots, n_m) = \sum_j f_j(n_j)$. These form of potentials includes the special case when the well-known Potts model is applied homogeneously on all edges of a clique. Let λ be the Potts potential of assigning two nodes of an edge the same label. The summation of these potentials over a clique is equivalent (up to a constant) to the clique potential $\mathcal{C}(n_1, \ldots, n_m) = \lambda \sum_j n_j^2$.

The Potts model with negative λ corresponds to the case when edges prefer the two end points to take different labels. With negative λ, our objective function $F(\mathbf{y})$ becomes concave and its maximum can be easily found using a relaxed quadratic program followed by an optimal rounding step as suggested in (Ravikumar & Lafferty, 2006). We therefore do not discuss this case further. The more interesting case is when λ is positive. We show that finding the optimal now becomes NP-hard.

Theorem 3.2. *When $\mathcal{C}(n_1, \ldots, n_m) = \lambda \sum_j n_j^2, \lambda > 0$, finding the MAP labeling is NP-hard.*

Proof. We show this by reducing from the NP-complete *exact cover by 3-sets* problem (Papadimitriou

& Steiglitz, 1982) of deciding if exactly $\frac{n}{3}$ of m subsets $S_1, \ldots, S_m$ of 3 elements each from $U = \{e_1, \ldots e_n\}$ can cover U. We let elements correspond to nodes and sets to labels. Assign $\phi_{ui} = 2n\lambda$ if $e_u \in S_i$ and 0 otherwise. MAP score will be $(2n^2 + 3^2\frac{n}{3})\lambda$ iff we can find an exact cover. □

The above proof establishes that there cannot be an algorithm that is polynomial in both n and m. But we have not ruled out algorithms with complexity that is polynomial in n but exponential in m, say of the form $O(2^m n^c)$ for a constant c.

We next propose approximation schemes. Unlike for general graphs where the Potts model is approximable only within a factor of $\frac{1}{2}$ (Boykov et al., 2001; Kleinberg & Tardos, 2002), we show that for cliques the Potts model can be approximated to within a factor of $\frac{13}{15} \approx 0.86$ using the α-pass algorithm of Section 3.1. We first present an easy proof for a weaker bound of $\frac{4}{5}$ and provide only a sketch of the more detailed proof for the $\frac{13}{15}$ bound. Let the optimal labeling be $\mathbf{y}^*$ and the labeling of α-pass be $\hat{\mathbf{y}}$.

Theorem 3.3. $F(\hat{\mathbf{y}}) \geq \frac{4}{5}F(\mathbf{y}^*)$.

Proof. Without loss of generality assume that the counts in $\mathbf{y}^*$ are $n_1 \geq n_2 \geq \ldots \geq n_m$. Then $\sum_i n_i^2 \leq n_1 \sum_i n_i = nn_1$.

$$\begin{aligned} F(\hat{\mathbf{y}}) &\geq F(\hat{\mathbf{y}}^{1n_1}) = \text{np}(\hat{\mathbf{y}}^{1n_1}) + \text{cp}(\hat{\mathbf{y}}^{1n_1}) \\ &\geq \text{np}(\mathbf{y}^*) + \text{cp}(\hat{\mathbf{y}}^{1n_1}) \quad \text{(from Claim 3.1)} \\ &\geq \text{np}(\mathbf{y}^*) + \lambda n_1^2 \quad \text{(since } \lambda > 0\text{)} \\ &\geq \text{np}(\mathbf{y}^*) + \text{cp}(\mathbf{y}^*) - \lambda n_1 n + \lambda n_1^2 \\ &\geq F(\mathbf{y}^*) - \lambda n^2/4 \end{aligned}$$

Now consider the two cases where $F(\mathbf{y}^*) \geq \frac{5}{4}\lambda n^2$ and $F(\mathbf{y}^*) < \frac{5}{4}\lambda n^2$. For the first case we get from above that $F(\hat{\mathbf{y}}) \geq F(\mathbf{y}^*) - \lambda n^2/4 \geq \frac{4}{5}F(\mathbf{y}^*)$. For the second case, we know that the score $F(\hat{\mathbf{y}}^{mn})$ where we assign all nodes the last label is at least λn^2 and thus $F(\hat{\mathbf{y}}) \geq \frac{4}{5}F(\mathbf{y}^*)$. □

Theorem 3.4. $F(\hat{\mathbf{y}}) \geq \frac{13}{15}F(\mathbf{y}^*)$.

Proof. (sketch) Without loss of generality, let $n_1 \geq n_2 \geq \ldots \geq n_k$ be k non-zero counts of labels in $\mathbf{y}^*$. When $k = 1$, i.e. a single label has all counts, α-pass will get the optimal. Consider cases where $k \geq 2$. The proof goes by contradiction. Let $F(\hat{\mathbf{y}})$ be a α-pass score where $F(\hat{\mathbf{y}}) < \frac{13}{15}F(\mathbf{y}^*)$. The following two hold for any $F(\hat{\mathbf{y}})$.

1. $F(\hat{\mathbf{y}}) \geq \text{np}(\mathbf{y}^*) + \lambda n_1^2$. We showed that this holds in the proof of 3.3.
2. $F(\hat{\mathbf{y}}) \geq \text{np}(\mathbf{y}^*)/k + \lambda n^2$. This holds because at least one of the k labels should have more than the average of the total node potential.

If we combine the above two with the inequality $F(\hat{\mathbf{y}}) < \frac{13}{15}F(\mathbf{y}^*)$, we get that $kn^2 - 13/2(k-1)\sum_{j=2}^{k} n_j^2 - n_1^2 < 0$. However, it can be shown that this cannot hold for any $n_1 \geq n_2 \geq \ldots \geq n_k$ where $k \geq 2$. □

Theorem 3.5. *The approximation ratio of $\frac{13}{15}$ of the α-pass algorithm is tight.*

Proof. We show an instance where this is obtained. Let $m = n + 3$ and $\lambda = 1$. For the first $n/3$ nodes let $\phi_{u1} = 4n/3$, for the next $n/3$ nodes let $\phi_{u2} = 4n/3$, and for the remaining $n/3$ let $\phi_{u3} = 4n/3$. Also for all nodes let $\phi_{u(u+3)} = 4n/3$. All other node potentials are zero. The optimal solution is to assign the first three labels $n/3$ nodes each, yielding a score of $4n^2/3 + 3(\frac{n}{3})^2 = 5n^2/3$. The first α-pass on label 1, where initially a node u is assigned its node optimal label $u+3$, will label the first $n/3$ nodes 1. This keeps the sum of total node potential unchanged at $4n^2/3$, the clique potential increased to $n^2/9 + 2n/3$ and total score $= 4n^2/3 + n^2/9 + 2n/3 = 13n^2/9 + 2n/3$. No subsequent α passes with any other label can improve this score. Thus, the score of α-pass is $\frac{13}{15}$ of the optimal in the limit. □

3.2.1. α-EXPANSION

In general graphs, a popular method that provides the approximation guarantee of 1/2 for the Potts model is the graph-cuts based α expansion algorithm. We explore the behavior of this algorithm.

In this scheme, we start with any initial labeling — for example, all nodes the first label as suggested in (Boykov et al., 2001). Next, for each label α we perform an α expansion phase where we switch the label of an optimal set of nodes to α from their current label. We repeat this until in a round over the m labels, no nodes switch their labels.

For graphs whose edge potentials form a metric, an optimal α expansion move is based on the use of the mincut algorithm of Boykov et al. (2001) which for the case of cliques can be $O(n^3)$.

We next show how to perform optimal α expansion moves more efficiently for all kinds of SUM potentials.

An α expansion move Let $\tilde{\mathbf{y}}$ be the labeling at the start of this move. For each label $y \neq \alpha$ create a sorted list S_y of nodes with label y in $\tilde{\mathbf{y}}$ in decreasing order of $\phi_{u\alpha} - \phi_{uy}$. If in an optimal move, we move k_y nodes from y to α, then it is clear that we need to pick the top k_y nodes from S_y. Let r_u be the rank

of a node u in S_y. Our remaining task is to decide the optimal number k_y to take from each S_y. We find these using dynamic programming. Without log of generality assume $\alpha = m$ and $1, \ldots, m-1$ are the $m-1$ labels other than α.

Let $D_i(k)$ denote the best score with k nodes from labels $1 \ldots i$ switched to α. We compute

$$D_i(k) = \max_{j \le k, j \le n_i(\tilde{\mathbf{y}})} D_{i-1}(k-j) + f_i(n_i(\tilde{\mathbf{y}}) - j) + \sum_{v: r_v \le j} \phi_{v\alpha} + \sum_{v: r_v > j} \phi_{vi}$$

From here we can calculate the optimal number of nodes to switch to α as $\text{argmax}_{j \le n - n_\alpha(\tilde{\mathbf{y}})} D_{m-1}(j) + f_\alpha(j + n_\alpha(\tilde{\mathbf{y}}))$.

Theorem 3.6. *The α-expansion algorithm provides no better approximation guarantee than 1/2 even for the special case of homogeneous Potts potential on cliques.*

Proof. Consider an instance where $m = k+1$, and $\lambda = 1$. Let $\phi_{u1} = 2n/k$ for all u and for k disjoint groups of n/k nodes each, let $\phi_{u,i+1} = 2n$ for the nodes in the i^{th} group. All other node potentials are zero. Consider the solution where every node is assigned label 1. This labeling is locally optimal wrt any α-expansion move, and its score is $n^2(1 + 2/k)$. However, the exact solution assigns every node group its label, with a score $n^2(2 + 1/k)$, thus giving a ratio of 1/2 in the limit. □

We next present a generalization of the α-pass algorithm that provides provably better guarantees while being faster than α-expansion.

3.2.2. Generalized α-pass algorithm

In α-pass for each label α, we go over each count k and find the best node score with k nodes labeled α. We generalize this to go over all label combinations of size no more than p, a parameter of the algorithm that is fixed based on the desired approximation guarantee.

For each label subset A of size no more than p, and for each count k, maximize node potentials with k nodes assigned a label from set A. For this, sort nodes in decreasing order of $\max_{\alpha \in A} \phi_{u\alpha} - \max_{y \notin A} \phi_{uy}$, assign the top k nodes their best label in A and the remaining their best label not in A. The best over all A, k with $|A| \le p$ is the returned labeling.

The complexity of this algorithm is $O(nm^p \log n)$. Since m is typically smaller than n, this is a useful option for large cliques. In practice, we can use heuristics to prune the number of label combinations. Further, we can make the following claims about the quality of its output. We skip the proof because it is quite involved.

Theorem 3.7. $F(\hat{\mathbf{y}}) \ge \frac{4p}{4p+1} F(\mathbf{y}^*)$.

The above bound is not tight as for $p = 1$ we have already shown that the $\frac{4}{5}$ bound can be tightened to $\frac{13}{15}$. With $p = 2$ we get a bound of $\frac{8}{9}$ which is better than $\frac{13}{15}$.

3.3. MAJORITY Potentials

MAJORITY potentials are of the form $\mathcal{C}(n_1, \ldots, n_m) = f_a(\mathbf{n})$, $a = \text{argmax}_y\ n_y$. We consider linear majority potentials where $f_a(\mathbf{n}) = \sum_y w_{ay} n_y$. The matrix $W = \{w_{yy'}\}$ need not be diagonally dominant or symmetric. We show that exact MAP for linear majority potentials can be found in polynomial time. We also present a modification to the α-pass algorithm to serve as an efficient approximation scheme.

3.3.1. Modified α-pass algorithm

In the case of linear majority potentials, we can incorporate the clique term in the node potential, and this leads to the following modifications to the α-pass algorithm: (a) Sort the list for α according to $\phi_{u\alpha} + w_{\alpha\alpha} - \max_{y \ne \alpha}(\phi_{uy} + w_{\alpha y})$, and (b) While sweeping the list for α, discard all candidate solutions whose majority label is not α.

However even after these modifications, α-pass does not provide the same approximate guarantee as for homogeneous Potts potentials. Infact, counter examples prove the following: (a) When W is unconstrained, the bound cannot be more than 1/2, and (b) Even if $\forall y, y'\ w_{yy} \ge w_{yy'}$, the bound cannot be better than 2/3.

However, in practice where the W matrix is typically sparse our experiments in Section 4 show that α-pass performs well and is significantly more efficient than the exact algorithm described next.

3.3.2. Exact Algorithm

One possible way to exactly solve for MAJORITY potentials is to guess the majority label α, the count $k = n_\alpha$, and solve the following IP:

$$\max_{\mathbf{y}} \sum_{u,y} (\phi_{uy} + w_{\alpha y}) x_{uy}$$
$$\forall y: \sum_u x_{uy} \le k,$$
$$\forall u: \sum_y x_{uy} = 1,\ x_{uy} \in \{0,1\} \qquad (2)$$

This IP solves the degree constrained bipartite matching problem, which can be solved exactly in polynomial time. Indeed, it can be easily shown that the LP relaxation of this IP has an integral solution. Thus we can solve $O(mn)$ such problems by varying α and k, and report the best solution. We believe that since the

subproblems are related, it should be possible to solve them incrementally using combinatorial approaches.

4. Experiments

In this section, we compare our algorithms against sequential tree re-weighted message passing (TRW-S) and graph-cut based inference for clique potentials that are decomposable over clique edges; and with ICM when the clique potentials are not edge decomposable. We compare them on running time and quality of the MAP. Our experiments were performed on both synthetic and real data.

Synthetic Dataset: We generated cliques with 100 nodes and 24 labels by choosing node potentials at random from $[0, 2]$ for all labels. A Potts version (POTTS) was created by gradually varying λ , and generating 25 cliques for every value of λ. We also created analogous ENTROPY, MAKESPAN and MAKESPAN2 versions of the dataset by choosing entropy, linear makespan ($\lambda \max_y n_y$) and square makespan ($\lambda \max_y n_y^2$) clique potentials respectively.

For majority potentials we generated two kinds of datasets (parameterized by λ): (a) MAJ-DENSE obtained by generating a random symmetric W for each clique, where $W_{ii} = \lambda$ was the same for all i and $W_{ij} \in [0, 2\lambda]$ $(j \neq i)$, and (b) MAJ-SPARSE from symmetric W with $W_{ij} \in [0, 2\lambda]$ for all i, j, roughly 70% of whose entries were zeroed.

Of these, only POTTS is decomposable over clique edges.

CoNLL Dataset: The CoNLL 2003 dataset[1] is a popular choice for demonstrating the benefit of collective labeling in named entity recognition tasks. We used the BIOU encoding of the entities, that resulted in 20 labels. We took a subset of 1460 records from the test set of CoNLL, and selected all 233 cliques of size 10 and above. The median and largest clique sizes were 16 and 259 respectively. The node potentials of the cliques were set by a sequential Conditional Random Field trained on a disjoint training set. We created a Potts version by setting $\lambda = 0.9/n$, where n is the clique size. Such a λ allowed us to balance the node and clique potentials for each clique. A majority version was also created by learning W in the training phase.

All our algorithms were written in Java. We compared these with C++ implementations of the TRW-S[2], and graph-cut based expansion algorithms[3] (Boykov et al., 2001; Szeliski et al., 2006; Kolmogorov & Zabih, 2004; Boykov & Kolmogorov, September 2004). All experiments were performed on a Pentium-IV 3.0 GHz machine with four processors and 2 GB of RAM.

[1] http://cnts.uia.ac.be/conll2003/ner/

[2] http://www.adastral.ucl.ac.uk/~vladkolm/papers/TRW-S.html

[3] http://vision.middlebury.edu/MRF/

4.1. Edge Decomposable Potentials

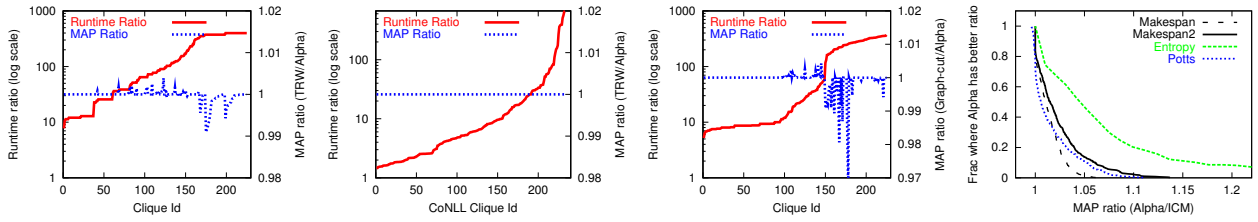
Figures 1(a) and 1(b) compare the performance of TRW-S vs α-pass on the two datasets. In Figure 1(a), we varied λ uniformly in $[0.8, 1.2]$ with increments of 0.05. This range of λ is of special interest, because it allows maximal contention between the clique and node potentials. For λ outside this range, the MAP is almost always a trivial labeling, viz. one which individually assigns each node to its best label, or assigns all nodes to a single label.

We compare two metrics — (a) the quality of the MAP score, captured by the ratio of the TRW-S MAP score with the α-pass MAP score, and (b) the runtime required to report that MAP, again as a ratio. Figure 1(a) shows that while both the approaches report almost similar MAP scores, the TRW-S algorithm is more than 10 times slower in more than 80% of the cases, and is never faster. This is expected because each iteration of TRW-S costs $O(n^2)$, and multiple iterations must be undertaken. In terms of absolute run times, a single iteration of TRW-S took an average of 193ms across all cliques in POTTS, whereas our algorithm returned the MAP in 27.6 ± 8.7ms. Similar behavior can be observed on CoNLL dataset in Figure 1(b). Though the degradation is not as much as before, mainly because of the smaller average clique size, TRW-S is more than 5 times slower on more than half the cliques.

Figure 1(c) shows the comparison with Graph-cut based expansion. The MAP ratio is even more in favor of α-pass, while the blowup in running time is of the same order of magnitude as TRW-S. This is surprising because based on the experiments in (Szeliski et al., 2006) we expected this method to be faster. One reason could be that their experiments were on grid graphs whereas ours are on cliques.

4.2. Non-decomposable Potentials

In this case, we cannot use the TRW or graph-cut based algorithms. Hence we compare with the ICM algorithm that has been popular in such scenarios (Lu & Getoor, 2003). We varied λ with increments of 0.02 in $[0.7, 1.1)$ and generated 500 cliques each from POTTS, MAJ-DENSE, MAJ-SPARSE, ENTROPY, MAKESPAN and MAKESPAN2. We measure the ratio of MAP score of α-pass with ICM and for each ratio r we plot the fraction of cliques where α-pass returns a MAP that results in a ratio better than r. Figure 1(d) shows the results on all the potentials except majority. The curves for



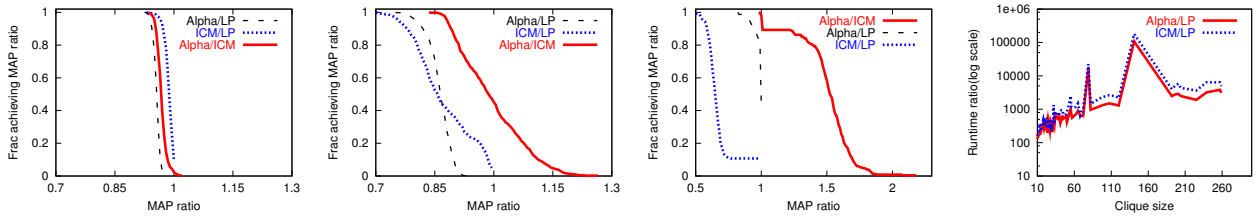
(a) Comparison with TRW-S: Synthetic

(b) Comparison with TRW-S: CoNLL

(c) Comparison with GraphCut: Synthetic

(d) Comparison with ICM: Synthetic

Figure 1. Comparison with TRW-S, Graph-cut and ICM



(a) MAJ-DENSE

(b) MAJ-SPARSE

(c) CoNLL

(d) Time(CoNLL)

Figure 2. Comparing AlphaPass, ICM and Exact on majority potentials

linear and square makespan lie totally to the right of $ratio = 1$, which is expected because α-pass will always return the optimal answers for those potentials. For Potts too, α-pass is better than ICM for almost all the cases. For entropy, α-pass was found to be significantly better than ICM in all the cases. The runtimes of ICM and α-pass were similar.

Majority Potentials

In Figures 2(a) and 2(b), we compare ICM and modified α-pass with the LP-based exact method, and there is no clear winner. The dotted curves plot, for each MAP ratio r, the fraction of cliques on which ICM (or α-pass) returns a MAP score better than r times the true MAP. The solid curve plots the fraction of cliques where α-pass returns a MAP score better than r times the ICM MAP. On MAJ-DENSE, both the algorithms return a MAP score better than 0.93 of the true MAP, with ICM being slightly better. However in MAJ-SPARSE, both the algorithms dominate on roughly half the cliques each. The MAP ratios returned are also over a wider range than MAJ-DENSE.

Figure 2(c) displays similar results on the CoNLL dataset, whose W matrix is 85% sparse. ICM performs poorly, returning the true MAP in only 10% of the cases across all clique sizes, and achieving an average MAP ratio of 0.68 against the exact method. On the other hand, α-pass returns the true MAP in more than 40% of the cases, with an excellent average MAP ratio of 0.98, and almost always provides a solution better than ICM.

In terms of runtime, ICM and α-pass are roughly 100-100000 times faster than the exact algorithm. Figure 2(d) displays these runtime ratios on all CoNLL cliques. Barring a few highly expensive outliers, and ignoring the dependence on m, it appears that the exact method is roughly $O(n)$ times slower than α-pass (and ICM). Thus, for practical majority potentials, α-pass seems to quickly provide highly accurate solutions.

An interesting problem however still remains: that of designing exact/approximate algorithms that can operate equally well on sparse and dense majority potentials, with runtimes similar to ICM and α-pass.

5. Concluding Remarks and Future Work

In this paper we identified real-life inference tasks on cluster graphs with large cliques whose potentials are based only on the cardinality of labels and single node

potentials. We exploit this special form of the clique potentials to design inference algorithms that provide provably good approximation guarantees, and are 1–2 orders of magnitude faster than state-of-the-art methods. We analyzed the case of the Potts model in depth and showed that the inference problem even in this simple case is NP-hard. We designed an approximation algorithm that provides a tight approximation ratio of $\frac{13}{15}$ in $O(mn \log n)$ time. This can be generalized to yield an approximation ratio of $\frac{4p}{4p+1}$ in $O(m^p n \log n)$ time in the worst case time.

As part of the future work, we would like to expand our set of algorithms to larger families of clique potential functions, design incremental algorithms for majority potentials and computing max-marginals within a clique, and perform inference on cluster graphs generated from real-life applications.

Acknowledgments The work reported here was supported by research grants from Microsoft Research and an IBM Faculty award.

References

Boykov, Y., & Kolmogorov, V. (September 2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *In IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (pp. 1124–1137).

Boykov, Y., Veksler, O., & Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.*, *23*, 1222–1239.

Bunescu, R., & Mooney, R. J. (2004). Collective information extraction with relational markov networks. *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics* (pp. 439–446).

Chakrabarti, S., Dom, B., & Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *SIGMOD Rec.*, *27*, 307–318.

Duchi, J., Tarlow, D., Elidan, G., & Koller, D. (2007). Using combinatorial optimization within max-product belief propagation. *Advances in Neural Information Processing Systems (NIPS 2006)*.

Finkel, J. R., Grenager, T., & Manning, C. D. (2005). Incorporating non-local information into information extraction systems by gibbs sampling. *ACL*.

Jensen, D., Neville, J., & Gallagher, B. (2004). Why collective inference improves relational classification. *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Kleinberg, J., & Tardos, E. (2002). Approximation algorithms for classification problems with pairwise relationships: metric labeling and markov random fields. *J. ACM*, *49*, 616–639.

Kolmogorov, V. (2004). *Convergent tree-reweighted message passing for energy minimization* (Technical Report MSR-TR-2004-90). Microsoft Research (MSR).

Kolmogorov, V., & Wainwright, M. J. (2005). On the optimality of tree-reweighted max-product message passing. *UAI*.

Kolmogorov, V., & Zabih, R. (2004). What energy functions can be minimized via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*.

Krishnan, V., & Manning, C. D. (2006). An effective two-stage model for exploiting non-local dependencies in named entity recognition. *ACL-COLING*.

Lu, Q., & Getoor, L. (2003). Link-based classification. *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA* (pp. 496–503).

Papadimitriou, C., & Steiglitz, K. (1982). *Combinatorial optimization: Algorithms and complexity*, chapter 11, 247–254. Prentice Hall, Englewood Cliffs, NJ.

Ravikumar, P., & Lafferty, J. (2006). Quadratic programming relaxations for metric labeling and markov random field map estimation. *ICML*.

Sutton, C., & McCallum, A. (2004). *Collective segmentation and labeling of distant entities in information extraction* (Technical Report TR # 04-49). University of Massachusetts. Presented at ICML Workshop on Statistical Relational Learning and Its Connections to Other Fields.

Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., & Rother, C. (2006). A comparative study of energy minimization methods for markov random fields. *In Ninth European Conference on Computer Vision (ECCV 2006)* (pp. 16–29).

Taskar, B., Abbeel, P., & Koller, D. (2002). Discriminative probabilistic models for relational data. *Eighteenth Conference on Uncertainty in Artificial Intelligence (UAI02), Edmonton, Canada*.

Taskar, B., Chatalbashev, V., & Koller, D. (2004). Learning associative markov networks. *Twenty First International Conference on Machine Learning (ICML04), Banff, Canada.*.