A Message Passing Algorithm for MRF Inference with Unknown Graphs and Its Applications

Zhenhua Wang[1,2], Zhiyi Zhang[2⋆], Nan Geng[2]

[1] School of Computer Science, The University of Adelaide
[2] College of Information Engineering, Northwest A&F University
zhhsun@outlook.com, {zhangzhiyi, nangeng}@nwsuaf.edu.cn

Abstract. Recent research shows that estimating labels and graph structures simultaneously in Markov random Fields can be achieved via solving LP problems. The scalability is a bottleneck that prevents applying such technique to larger problems such as image segmentation and object detection. Here we present a fast message passing algorithm based on the mixed-integer bilinear programming formulation of the original problem. We apply our algorithm to both synthetic data and real-world applications. It compares favourably with previous methods.

1 Introduction

Many computer vision applications involve predicting structured labels like sequence and trees. A potential function is typically defined to measure the consistency between structured label candidates and observations, and maximising the potential function over the labelling space discloses the structured label estimation. An example is the semantic image segmentation (pixel labelling) task which requires assigning each pixel or superpixel a label representing the corresponding object category. Labels of all pixels form a sequence. A typical potential function for this task is a sum of all unary potentials and pairwise potential potentials, where each unary term measures the consistency between the pixel label and the photometric information of the pixel, and each pairwise term evaluates the consistency between the labels of neighbouring pixels [1].

Markov random fields (MRFs) provide a compact representation of the dependency among structured variables. Each random variable is typically represented by a node in the MRF graph, and the dependency between a pair of variables is encoded by an edge in the graph. A vacancy of edge between two nodes indicates that the associated variables are independent conditioned on observing the statuses of all rest variables. In this sense, the graph structure of MRF is essential in modelling the structured prediction problems. Despite maximising the potential function is NP-hard in general, approximations can be found efficiently by carrying out message passing [2] on MRF graphs.

⋆ Corresponding author

To determine the structure of MRF graphs, one usually chooses to optimise the information gain [3]. Alternatively, rules based on heuristics or domain knowledge can be used. For example, in image segmentation, one usually uses grid graphs with edges reflecting pixel adjacency; in human activity recognition with multiple persons, one can use tree structured graphs that span shortest Euclidean distances across all people. However, determining graphs heuristically or based on domain knowledge is not principle and is prone to input variance. First of all, if we create graphs by defining rules derived from domain knowledge, the rules are always too problem-oriented to be generally applicable. Second, unwanted edges can be easily introduced by using heuristics. For example, in human activity recognition, graphs generated according to the near-far relationship between people can be undesirable because two persons might be interacting even when they are far away from each other (passing basketball for instance). Due to these reasons, digging adaptive graphs directly from inputs is interesting.

Inferring graphs and labels directly and simultaneously from data has shown to be favourable comparing with using fixed hand-engineered graphs in human action recognition [4, 5]. However, the related inference problem is highly challenging. Lan *et al.* [4] propose an approach to the problem which alternates between finding the best label for a fixed graph using loopy belief propagation, and finding the best graph for a fixed set of labels via solving a LP. A rounding scheme is used to decode the structures from LP solutions. Recently, Wang *et al.* [5] showed that the problem of finding labels and graphs jointly can be formulated as a bilinear programming (BLP) problem, which they then relaxed to a LP problem. A branch and bound (B&B) method was then developed to improve the quality of the solution using the LP as bounds, which essentially involves solving a number of LPs. Unfortunately, this B&B method is extremely time-consuming even for small graphs, meaning that an early-stop is usually used which results in a sub-optimal solution. To enable inferring graphs and labels simultaneously on large-scale problems, we propose a message passing-style algorithm in this paper. We formulate the inference as a *mixed integer bilinear programming problem* [6]. Then we derive the *partial-dual* (the term is probably first used in [7]) of the mixed integer bilinear programming problem. To solve the dual problem, we fix a majority of variables in the partial-dual and the reduced problem can be solved *analytically*. This approach can be viewed as a message passing process which extends Globerson's MPLP algorithm [8] to MRF inference with unknown graphs. We apply our algorithm to both synthetic data and real computer vision tasks including semantic image segmentation and human action recognition. Our algorithm is competitive with the state-of-the-art on accuracy while is much faster.

The rest of the paper is organised as follows. In Section 2 we show our formulation of the inference problem. Next in Section 3 we describe our message passing algorithm. Then we compare our algorithm with other methods on synthetic data in Section 4. Finally in Section 5 we show the applications of our algorithm on semantic image segmentation and human activity recognition.

2 Mixed Integer Bilinear Programming Formulation

Let $\mathcal{V} = \{1, 2, \dots, n\}$ be the node set; $\mathcal{E} = \{(i,j)|i,j \in \mathcal{V}, i < j\}$ be the set containing all possible edges; $y_i \in \mathcal{Y}$ denote the discrete random variable corresponding to node i; and $\mathbf{y} = [y_i]_{i \in \{1,2,\dots,n\}}$ be a collective representation of all random variables. Introducing binary variables $z_{ij} \in \{0,1\}, \forall (i,j) \in \mathcal{E}$ to represent if an edge (i,j) exist ($z_{ij} = 1$) or not ($z_{ij} = 0$), and letting $\mathbf{z} = [z_{ij}]_{i,j \in \mathcal{V}, i<j}$ be a collective representation of all z_{ij} variables formed by collecting all z_{ij} variables in the order of enumerating all possible i and j indices in turn. Following [5], inferring graphs and labels simultaneously can be formulated as the following:

$$\begin{aligned} &\max_{\mathbf{y},\mathbf{z}} \sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) z_{ij}, \\ &\text{s.t.} \sum_{(i,j) \in \mathcal{E}} \mathbb{1}(i = k \text{ or } j = k) z_{ij} \le h, \forall k \in \mathcal{V}\,. \end{aligned} \tag{1}$$

Here $\theta_i(y_i), \theta_{ij}(y_i, y_j)$ denote unary and pairwise potentials respectively, $\mathbb{1}(\cdot)$ is an indicator function that gives 1 if the condition inside the brackets is true, and gives 0 otherwise. The constraints control the sparsity of the estimated graph by enforcing the maximum degree of the graph less than a constant h. When $\{z_{ij}\}_{(i,j) \in \mathcal{E}}$ is given, *i.e.* we known the graph structure, the above problem recovers the traditional MRF inference problem.

Formulation. Introducing binary variables $\mu_i(y_i) \in \{0,1\}\ \forall i \in \mathcal{V}$, and binary variables $\mu_{ij}(y_i, y_j)\ \ \forall (i,j) \in \mathcal{E}, y_i, y_j$. Let $\boldsymbol{\mu}_1 = [\mu_i(y_i)]_{i \in \mathcal{V}, y_i \in \mathcal{Y}}$, $\boldsymbol{\mu}_2 = [\mu_{ij}(y_i, y_j)]_{i<j, y_i, y_j \in \mathcal{Y}}$ be the collective representations of all $\mu_i(y_i), \mu_{ij}(y_i, y_j)$ variables respectively by collecting all $\mu_i(y_i)$ and $\mu_{ij}(y_i, y_j)$ variables in the order of enumerating all possible $i, j \in \mathcal{V}, y_i, y_j \in \mathcal{Y}$ in turn. Problem (1) can be equivalently written as

$$\begin{aligned} \max_{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \mathbf{z}} \ & \sum_{i \in \mathcal{V}} \sum_{y_i} \mu_i(y_i) \theta_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) \theta_{ij}(y_i, y_j) z_{ij} \\ \text{s.t.} \ & \sum_{y_i} \mu_i(y_i) = 1\ \forall i \in \mathcal{V}, \\ & \sum_{y_i, y_j} \mu_{ij}(y_i, y_j) = 1\ \forall (i,j) \in \mathcal{E}, \\ & \sum_{y_i} \mu_{ij}(y_i, y_j) = \mu_j(y_j)\ \forall (i,j) \in \mathcal{E}, y_j, \\ & \sum_{y_j} \mu_{ij}(y_i, y_j) = \mu_i(y_i)\ \forall (i,j) \in \mathcal{E}, y_i, \\ & \sum_{(i,j) \in \mathcal{E}} \mathbb{1}(i = k \text{ or } j = k) z_{ij} \le h, \forall k \in \mathcal{V}, \end{aligned} \tag{2}$$

which can be relaxed into a mixed integer bilinear programming problem:

$$\begin{aligned} \max_{\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\mathbf{z}} \quad & \sum_{i\in\mathcal{V}}\sum_{y_i}\mu_i(y_i)\theta_i(y_i) + \sum_{(i,j)\in\mathcal{E}}\sum_{y_i,y_j}\mu_{ij}(y_i,y_j)\theta_{ij}(y_i,y_j)z_{ij} \\ \text{s.t.} \quad & (\boldsymbol{\mu}_1,\boldsymbol{\mu}_2,\mathbf{z}) \in \mathcal{M}, \end{aligned} \tag{3}$$

where $\mathcal{M}$ is a space defined as

$$\mathcal{M} = \left\{ \boldsymbol{\mu},\mathbf{z} \;\middle|\; \begin{array}{l} \sum_{y_i}\mu_i(y_i) = 1, \forall i \in \mathcal{V}, \\ \sum_{y_i,y_j}\mu_{ij}(y_i,y_j) = 1, \forall (i,j) \in \mathcal{E}, \\ \sum_{y_i}\mu_{ij}(y_i,y_j) = \mu_j(y_j), \forall (i,j) \in \mathcal{E}, y_j, \\ \sum_{y_j}\mu_{ij}(y_i,y_j) = \mu_i(y_i), \forall (i,j) \in \mathcal{E}, y_i, \\ \sum_{(i,j)\in\mathcal{E}} \mathbb{1}(i = k \text{ or } j = k)z_{ij} \le h, \forall k \in \mathcal{V}, \\ \mu_i(y_i) \in [0,1], \forall i \in \mathcal{V}, y_i, \\ \mu_{ij}(y_i,y_j) \in [0,1], \forall (i,j) \in \mathcal{E}, y_i, y_j, \\ z_{ij} \in \{0,1\}, \forall (i,j) \in \mathcal{E}\,. \end{array} \right\} \tag{4}$$

Note our mixed integer bilinear formulation is exactly the same as the bilinear relaxation in [5] except that $z_{ij} \in \{0,1\}$ in our problem as compared with $z_{ij} \in [0,1]$ in the bilinear formulation in [5]. As a result, our relaxation (3) is tighter than the bilinear relaxation. As we will see later, the formulation leads to not only very fast algorithm scales to large inference problem, but also the closed-form solution for updating beliefs.

3 The Message Passing Algorithm

Message passing, also known as belief propagation is a strategy to perform inference on probabilistic graphical models, *e.g.* MRFs. The success of message passing algorithms lies in splitting the original inference problem into small sub-problems according to the structure of the problem (known as factorisation), where each sub-problem can be efficiently solved via propagating messages among nodes.

Compared with traditional message passing algorithms performed on MRF graphs with known structures, our message passing algorithm has two differences. Firstly, it does not require knowing the graph structure of MRF. Instead, the algorithm automatically estimates the graph structure and labelling simultaneously in a unique framework. Secondly, we derive a partial-dual of the original inference problem and perform the messaging passing in the partial-dual space. In comparison with existing algorithms for MRF inference with unknown graphs, our algorithm is significantly faster because it iteratively solves the sub-problems of the partial-dual problem which have analytical solutions.

3.1 The partial-dual problem

It turns out that the following problem is equivalent to the problem (3):

$$\begin{aligned}
&\max_{\mathbf{z}} \quad \min_{\boldsymbol{\beta}} \sum_{i\in\mathcal{V}} \max_{y_i} \sum_{j\in\mathcal{V}\setminus\{i\}} \max_{y_j} \beta_{ji}(y_j, y_i) + \theta_i(y_i) \\
&\text{s.t.} \quad \textstyle\sum_{(i,j)\in\mathcal{E}} \mathbb{1}(i = k \text{ or } j = k) z_{ij} \le h, \forall k \in \mathcal{V}, \\
&\qquad z_{ij} \in \{0, 1\} \ \forall (i, j) \in \mathcal{E}, \\
&\qquad \beta_{ij}(y_i, y_j) + \beta_{ji}(y_j, y_i) = \theta_{ij}(y_i, y_j) z_{ij} \ \forall (i, j) \in \mathcal{E}, y_i, y_j. \qquad (5)
\end{aligned}$$

Remember $\mathbf{z}$ are the primal variables that represent the graph structure. Here $\boldsymbol{\beta} = [\beta_{ij}(y_i, y_j)]_{i\neq j,y_i,y_j}$ are the dual variables. Despite the existence of the primal variables $\mathbf{z}$, for a fixed $\mathbf{z}$ this problem is called the partial-dual problem of (3) because it is actually a Lagrangian dual of the primal problem (3) when $\mathbf{z}$ are known.

The derivation is briefed as follows. First we fix all structure variables $\mathbf{z}$ and the problem (3) becomes a linear programming problem of $\boldsymbol{\mu}$, for which we next derive a Lagrangian dual using the technique presented by Globerson *et al.* in [8]. Then we remove redundant constraints and variables, leaving only the dual variables $\boldsymbol{\beta}$. At last $\mathbf{z}$ are reset as free variables and we get (5). In comparison with the primal version (3), the dual problem contains far fewer constraints. However, solving such a problem is still difficult. We next present our message passing algorithm that solves (5) approximately but efficiently.

3.2 The algorithm

In order to solve (5), we adopt an iterative strategy. Concisely, during each iteration we fix all the primal and dual variables in (5) except for the variables related to one selected edge. The reduced problem is solved analytically. This process is repeated until a max-number of iterations is reached.

Problem reduction. Let $\mathcal{E}^*$ denote the current edge estimation, and let $\mathbf{z}^*$ denote the corresponding solution of structure variables. During each iteration a node pair (i, j) is selected. By fixing all variables unchanged except for variables related to (i, j), *i.e.* z_{ij}, $\beta_{ij}(y_i, y_j)$ and $\beta_{ji}(y_j, y_i)$ $\forall y_i, y_j$, the problem (5) becomes

$$\begin{aligned}
&\max_{z_{ij}\in\{0,1\}} \quad \min_{\boldsymbol{\beta}_{ij},\boldsymbol{\beta}_{ji}} q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) \\
&\text{s.t. } \beta_{ij}(y_i, y_j) + \beta_{ji}(y_j, y_i) = \theta_{ij}(y_i, y_j) z_{ij} \ \forall y_i, y_j, \\
&\sum_{(r,s)\in\mathcal{E}^*} \mathbb{1}(r = k \text{ or } s = k) z^*_{rs} - z^*_{ij} + z_{ij} \le h \ \forall k \in \{i, j\}, \\
&\beta_{ij}(y_i, y_j), \beta_{ji}(y_j, y_i) \in [0, 1] \ \forall y_i, y_j. \qquad (6)
\end{aligned}$$

Here $\boldsymbol{\beta}_{ij} = [\beta_{ij}(y_i, y_j)]_{\forall y_i, y_j}, \boldsymbol{\beta}_{ji} = [\beta_{ji}(y_j, y_i)]_{\forall y_j, y_i}$, and the objective function

$$\begin{aligned} q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) = &\max_{y_i}[\lambda_i^{-j}(y_i) + \max_{y_j} \beta_{ji}(y_j, y_i) + \theta_i(y_i)] + \\ &\max_{y_j}[\lambda_j^{-i}(y_j) + \max_{y_i} \beta_{ij}(y_i, y_j) + \theta_j(y_j)], \end{aligned} \tag{7}$$

where λ_i^{-j}, λ_j^{-i} are compact representations of the following:

$$\lambda_i^{-j}(y_i) = \textstyle\sum_{k \in \mathcal{V} \backslash \{i,j\}} z_{ki}^* \max_{y_k} \beta_{ki}(y_k, y_i), \tag{8}$$

$$\lambda_j^{-i}(y_j) = \textstyle\sum_{k \in \mathcal{V} \backslash \{i,j\}} z_{kj}^* \max_{y_k} \beta_{kj}(y_k, y_j). \tag{9}$$

As in [8], we define

$$\lambda_{ki}(y_i) = \max_{y_k} \beta_{ki}(y_k, y_i) \tag{10}$$

as the message passing from node k to node i. According to (8) and (9), $\lambda_i^{-j}(y_i)$ is an accumulation of messages passing from all neighbouring nodes (except for j) to i when it takes the label y_i, and $\lambda_j^{-i}(y_j)$ is an accumulation of messages passing from all neighbouring nodes (except for i) to j when it takes the label y_j. As we will see later, these messages carry essential information needed for updating the current solutions.

Update via message passing. Because z_{ij} is a binary variable, we choose to exhaustively search over $z_{ij} \in \{0, 1\}$. We have the following proposition:

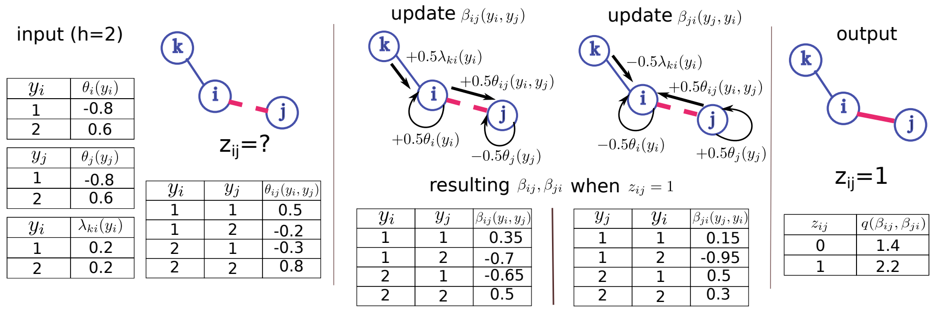
Proposition 1. *For any particular z_{ij}, the problem (6) actually has analytical solutions: minimising $q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji})$ yields the following results*

$$\beta_{ij}(y_i, y_j) = \tfrac{1}{2}[\lambda_i^{-j}(y_i) + \theta_{ij}(y_i, y_j) z_{ij} + \theta_i(y_i) - \lambda_j^{-i}(y_j) - \theta_j(y_j)], \tag{11}$$

$$\beta_{ji}(y_j, y_i) = \tfrac{1}{2}[\lambda_j^{-i}(y_j) + \theta_{ij}(y_i, y_j) z_{ij} + \theta_j(y_j) - \lambda_i^{-j}(y_i) - \theta_i(y_i)]. \tag{12}$$

Proof. Let $\hat{z}_{ij}$ denote a fixed value of z_{ij}. According to Equation (7), the following inequality holds:

$$\begin{aligned} q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) \geq &\max_{y_i, y_j}\{\lambda_i^{-j}(y_i) + \lambda_j^{-i}(y_j) + \beta_{ji}(y_j, y_i) + \beta_{ij}(y_i, y_j) + \theta_i(y_i) + \theta_j(y_j)\} = \\ &\underbrace{\max_{y_i, y_j}\{\lambda_i^{-j}(y_i) + \lambda_j^{-i}(y_j) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \theta_i(y_i) + \theta_j(y_j)\}}_{LB}. \end{aligned} \tag{13}$$


input (h=2)
k
i
j
z_{ij}=?
update $\beta_{ij}(y_i, y_j)$
$+0.5\lambda_{ki}(y_i)$
$+0.5\theta_{ij}(y_i, y_j)$
$+0.5\theta_i(y_i)$
$-0.5\theta_j(y_j)$
update $\beta_{ji}(y_j, y_i)$
$-0.5\lambda_{ki}(y_i)$
$+0.5\theta_{ij}(y_i, y_j)$
$-0.5\theta_i(y_i)$
$+0.5\theta_j(y_j)$
output
$z_{ij}=1$
resulting β_{ij}, β_{ji} when $z_{ij} = 1$

y_i	$\theta_i(y_i)$
1	-0.8
2	0.6

y_j	$\theta_j(y_j)$
1	-0.8
2	0.6

y_i	$\lambda_{ki}(y_i)$
1	0.2
2	0.2

y_i	y_j	$\theta_{ij}(y_i, y_j)$
1	1	0.5
1	2	-0.2
2	1	-0.3
2	2	0.8

y_i	y_j	$\beta_{ij}(y_i, y_j)$
1	1	0.35
1	2	-0.7
2	1	-0.65
2	2	0.5

y_j	y_i	$\beta_{ji}(y_j, y_i)$
1	1	0.15
1	2	-0.95
2	1	0.5
2	2	0.3

z_{ij}	$q(\beta_{ij}, \beta_{ji})$
0	1.4
1	2.2

Fig. 1: Updating z_{ij} via message passing for a toy example with three nodes {i,j,k}. Here $h = 2$. The required information includes potentials $\theta_i(y_i), \theta_j(y_j), \theta_{ij}(y_i, y_j) \forall y_i, y_j$, and the messages propagated from all other nodes to i, j, *i.e.* $\lambda_{ki}(y_i) \forall y_i$. The middle diagram visualise the computation of $\beta_{ij}(y_i, y_j)$ and $\beta_{ji}(y_j, y_i)$. Arrows denote message or potential flows when $z_{ij} = 1$. The function values of $q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji})$ are given by the table at the bottom of the right diagram. The value of z_{ij} is the one in {0,1} that gives larger $q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji})$ and does not violate any sparsity constraints in (1).

Hence LB is a lower bound of $q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji})$. Plug the $\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}$ given (11) and (12) into Equation (7), we have:

$$\begin{aligned} q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) = \max_{y_i, y_j} [\lambda_i^{-j}(y_i) + \frac{1}{2} \max_{y_j} (\lambda_j^{-i}(y_j) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \theta_j(y_j) - \lambda_i^{-j}(y_i) - \\ \theta_i(y_i)) + \theta_i(y_i)] + \max_{y_i, y_j} [\lambda_j^{-i}(y_j) + \frac{1}{2} \max_{y_i} (\lambda_i^{-j}(y_i) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \\ \theta_i(y_i) - \lambda_j^{-i}(y_j) - \theta_j(y_j)) + \theta_j(y_j)]. \end{aligned} \tag{14}$$

$$\begin{aligned} \implies q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) = \\ \max_{y_i, y_j} [\lambda_i^{-j}(y_i) + \frac{1}{2} (\lambda_j^{-i}(y_j) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \theta_j(y_j) - \lambda_i^{-j}(y_i) - \theta_i(y_i)) + \theta_i(y_i)] + \\ \max_{y_i, y_j} [\lambda_j^{-i}(y_j) + \frac{1}{2} (\lambda_i^{-j}(y_i) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \theta_i(y_i) - \lambda_j^{-i}(y_j) - \theta_j(y_j)) + \theta_j(y_j)]. \end{aligned} \tag{15}$$

$$\implies q(\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}) = \max_{y_i, y_j} \{\lambda_i^{-j}(y_i) + \lambda_j^{-i}(y_j) + \theta_{ij}(y_i, y_j)\hat{z}_{ij} + \theta_i(y_i) + \theta_j(y_j)\}, \tag{16}$$

which means that LB can be reached with $\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}$ given by (11) and (12). Since we are minimising the objective in (6) over $\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}$, the proof is complete.

□

Note when $z_{ij} = 0$, setting $\beta_{ij}(y_i, y_j) = 0$, $\beta_{ji}(y_j, y_i) = 0$ also solves the optimisation problem (6). In such case, we just use this trivial solution due to two reasons. First, the computation of (11) and (12) can be avoided. Second, this trivial solution gives zeros messages between (i, j), which is coherent with the fact that there is no edge between node i and node j.

Let $\{\beta^0_{ij}, \beta^0_{ji}\}$, $\{\beta^1_{ij}, \beta^1_{ji}\}$ denote the solution of (6) when z_{ij} equals 0 and 1 respectively. To get the final solution, we need to know the optimal value of z_{ij}. Since we are maximising over z_{ij}, the optimal z_{ij} is 1 if two conditions are met: 1) $q(\beta^0_{ij}, \beta^0_{ji}) < q(\beta^1_{ij}, \beta^1_{ji})$; 2) all sparsity constraints in (6) are not violated when setting $z_{ij} = 1$. Otherwise we let $z_{ij} = 0$. Updating z_{ij} via this method is illustrated in Figure 1. If the optimal value of z_{ij} is 1, we compute $\beta_{ij}(y_i, y_j), \beta_{ji}(y_j, y_i)$ according to (12); otherwise we set $\beta_{ij}(y_i, y_j), \beta_{ji}(y_j, y_i)$ to 0. Then we can update messages $\lambda_{ij}(y_j)$ and $\lambda_{ji}(y_i)$ according to (10). Note in practice, it is not necessary to store β values explicitly as all information we need for further computation is included in messages. During each iteration, we randomly select an edge (i, j) and solve the associated problem (6) exactly. Then we evaluate the objective in (1) using the current solution as inputs. If the current solution improves this objective, it is kept otherwise we discard it and consider the next (i, j). As shown in Figure 1, computing z_{ij} and $\{\boldsymbol{\beta}_{ij}, \boldsymbol{\beta}_{ji}\}$ can be viewed as a process of passing messages to nodes i and j from other nodes. Hence we call this algorithm partial-dual based message passing (PDMP). More details about our algorithm can be found in Algorithm 1. Note the decoding, *i.e.* determining the labelling $\mathbf{y}$ is achieved via maximising the so-called node beliefs over the labelling space of each node.

Currently the PDMP algorithm supports pairwise potentials only. However, with a modification of the sparsity constraints (*e.g.* restricting the total number of super-edges), a similar message passing algorithm for graphs with arbitrary cliques can be obtained.

4 Running Time Comparison

We compare the running time of our PDMP algorithm against the following methods:

- Lan the method proposed in [4] which alternatingly implement two steps: 1) fix graph structure and solve a MRF inference problem (with known graph); 2) fix labels and solve a LP problem. See [4] for more details.
- LP solves a linear programming relaxation [5] of the inference problem (2). The LP problems are solved using the Mosek toolbox [9].
- LP+B&B the branch and bound method proposed in [5]. The bounds are computed via solving the LP relaxation.

We generate synthetic data using a method similar to [10]. The node potentials are uniformly sampled from $\mathcal{U}(-1, 1)$, while the edge potentials are created as a product of a coupling strength and a distance $\text{dis}(y_i, y_j)$ between labels y_i, y_j. The coupling strength is sampled from $\mathcal{U}(-1, 1)$. Four types of distance functions are used including linear: $\text{dis}(y_i, y_j) = |y_i - y_j|$, quadratic: $\text{dis}(y_i, y_j) = (y_i - y_j)^2$, Ising: $\text{dis}(y_i, y_j) = y_i y_j$, Potts: $\text{dis}(y_i, y_j) = \mathbb{1}(y_i = y_j)$. We compare the average running time of solving twenty different synthetic examples with the number of nodes fixed to be 30. We let the sparsity parameter $h = 2$. The results are shown in Table 1. Clearly our PDMP algorithm is the fastest, while the LP+B&B is the slowest.

Algorithm 1 PDMP Algorithm.

```
Require: potentials $\theta$, $h$, max iteration number $tmax$.
Output: estimated $\mathbf{y}^*$ and $\mathbf{z}^*$.
1: Initialise: $\lambda_{ij}(y_j) \leftarrow 0$, $\lambda_{ji}(y_i) \leftarrow 0$, $z_{ij} \leftarrow 0$, $t \leftarrow 0$, $o_t \leftarrow -\infty$.
2: while $t < tmax$ do
3:     for each $(i,j) \in \mathcal{E}$ (pick $(i,j)$ randomly without repetition) do
4:         compute $\beta^1_{ij}(y_i, y_j)$, $\beta^1_{ji}(y_j, y_i)$ via (12).
5:         $\beta^0_{ij}(y_i, y_j) \leftarrow 0$, $\beta^0_{ji}(y_j, y_i) \leftarrow 0$, $z_{ij} \leftarrow 1$.
6:         if $q(\boldsymbol{\beta}^0_{ij}, \boldsymbol{\beta}^0_{ji}) < q(\boldsymbol{\beta}^1_{ij}, \boldsymbol{\beta}^1_{ji})$ and $\mathbf{z}$ is feasible then
7:             $\beta_{ij}(y_i, y_j) \leftarrow \beta^1_{ij}(y_i, y_j)$, $\beta_{ji}(y_j, y_i) \leftarrow \beta^1_{ji}(y_j, y_i)$.
8:         else
9:             $\beta_{ij}(y_i, y_j) \leftarrow \beta^0_{ij}(y_i, y_j)$, $\beta_{ji}(y_j, y_i) \leftarrow \beta^0_{ji}(y_j, y_i)$, $z_{ij} \leftarrow 0$.
10:        end if
11:        update messages $\lambda_{ij}(y_j)$, $\lambda_{ji}(y_i)$ via (10).
12:    end for
13:    compute node beliefs: $b_i(y_i) \leftarrow \sum_{k \in \mathcal{V} \setminus \{i\}} z_{ki} \lambda_{ki}(y_i) + \theta_i(y_i)$.
14:    decode: $\mathbf{y} \leftarrow [y_i]$ with $y_i \leftarrow \max_{y_i} b_i(y_i)$.
15:    $o_{t+1} \leftarrow \sum_{i \in \mathcal{V}} \theta_i(y_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(y_i, y_j) z_{ij}$.
16:    if $o_t < o_{t+1}$ then
17:        $\mathbf{y}^* \leftarrow \mathbf{y}$, $\mathbf{z}^* = [z_{uv}]$ $\forall (u,v) \in \mathcal{E}$.
18:    end if
19:    $t \leftarrow t + 1$.
20: end while
21: return $\mathbf{z}^*$ and $\mathbf{y}^*$.
```

5 Applications

We apply the proposed method to semantic image segmentation and human activity recognition. To our knowledge, this is the first work that estimates labels and graph structures simultaneously in semantic image segmentation.

5.1 Semantic Image Segmentation

Given an over-segmented image, the task here is to assign each super-pixel in the over-segmentation a label to express its object category.

A number of datasets are publicly available. In this paper we use the KITTI dataset [11]. The original dataset contains both 2D images (1240×380) and 3D laser data taken by a vehicle in different urban scenes. Since 2D information is more general in practice, we discard 3D information in our experiments.

There are 70 labelled images made by [12] as groundtruth. The original labelling contains 10 classes: road, building, vehicle, people, pavement, vegetation, sky, signal, post/pole and fence. As in [13], the 10 classes are mapped to five more general classes that are: ground (road and pavement), building, vegetation, and objects (vehicle, people, signal, pole and fence). Following [13, 12], the labelled images are divided into two parts containing 45 and 25 images respectively. The first part is used for training while the second part is used for testing.

	Ising	Linear	Quadratic	Potts
LP	17	12	11	11
Lan [4]	2	1	1	1
LP+B&B [5]	570	402	403	403
PDMP (ours)	**0.01**	**0.007**	**0.007**	**0.007**

Table 1: Comparison of running time (by seconds) on synthetic data generated by using different distance functions. For each distance function the best is highlighted.

	$\phi_2(y_i, y_j)$	$\phi_3(y_i, y_j)$	G	inference
CRF–Adj	$\exp(-\parallel \mathbf{c}_i - \mathbf{c}_j \parallel_2)$ if $y_i = y_j$, $1 - \exp(-\parallel \mathbf{c}_i - \mathbf{c}_j \parallel_2)$ if $y_i \neq y_j$	$\exp(-\parallel \mathbf{q}_i - \mathbf{q}_j \parallel_2)$ if $y_i = y_j$, $1 - \exp(-\parallel \mathbf{q}_i - \mathbf{q}_j \parallel_2)$ if $y_i \neq y_j$	Adj	BP
CRF–MST	same as CRF–Adj	same as CRF–Adj	2D MST	BP
Lan	$-\log(\parallel \mathbf{c}_i - \mathbf{c}_j \parallel_2)$ if $y_i = y_j$, 0 if $y_i \neq y_j$	$-\log(\parallel \mathbf{q}_i - \mathbf{q}_j \parallel_2)$ if $y_i = y_j$, 0 if $y_i \neq y_j$	Lan [4]	Lan [4]
PDMP	same as Lan	same as Lan	PDMP	PDMP

Table 2: Methods for image segmentation. The column G gives the approach used to create graph structures, and the column *inference* lists the methods used to solve the inference problem. Here BP means belief propagation, *Adj* stands for *Adjacent*. Note the first two methods can also use the $-\log$ potential functions used by Lan and PDMP. However, the performance is not as good as using the exp potentials.

A MRF based image segmentation strategy is adopted here. Each image $\mathbf{x}$ is over-segmented into small regions (super-pixels) at first using SLIC toolbox [14]. The super-pixels and their relations are represented by a graph $G = (\mathcal{V}, \mathcal{E})$ with the edge set $\mathcal{E}$ *unknown*. Each node $i \in \mathcal{V}$ in the MRF graph denotes a label y_i of the related super-pixel i. Each edge $(i, j) \in \mathcal{E}$ in the MRF graph encodes the dependency between the associated labels y_i, y_j. Let $\phi_1(y_i), \phi_2(y_i, y_j), \phi_3(y_i, y_j)$ denote the node feature, the edge feature in relevant to colour, and the edge feature in relevant to super-pixel location respectively. The potential function (parameterised by $\mathbf{w} = [w_1, w_2, w_3]$) is

$$F(\mathbf{x}, \mathbf{w}; \mathbf{y}, G) = \sum_{i \in \mathcal{V}} w_1 \phi_1(y_i) + \sum_{(i,j) \in \mathcal{E}} w_2 \phi_2(y_i, y_j) + w_3 \phi_3(y_i, y_j). \tag{17}$$

Maximising F over $\mathbf{y}$ and G uncovers the label estimation of all super-pixels. We test four methods: 1) CRF–Adjacency (Adj); 2) CRF–Minimum spanning tree (MST); 3) Lan; 4) PDMP. These methods are summarised by Table 2.

Features. To compute the node feature ϕ_1, we use the method employed in [13]: for each super-pixel, image features are extracted; then a classifier is trained on the extracted features; with the trained classifier, a score vector is computed for each super-pixel with each score representing the confidence of labelling the super-pixel by a particular label candidate. Let $\mathbf{c}_i$, $\mathbf{q}_i$ denote the LAB colour, the 2D position of the super-pixel i respectively. The definitions of ϕ_2 and ϕ_3 for different methods are given in Table 2. For Lan and PDMP methods, in order

	ground	objects	building	vegetation	sky	overall	mean	time
CRF–Adj	96.3%	63.9%	**87.7**%	90.5%	91.3%	84.4%	85.9%	0.516
CRF–MST	96.3%	67.8%	84.6%	**96.5**%	97.7%	86.2%	88.6%	**0.023**
Lan [4]	**97.9**%	71.3%	83.7%	87.6%	97.4%	85.1%	87.6%	7.323
PDMP (ours)	97.6%	**73.1**%	87.3%	95.1%	**98.3**%	**88.3**%	**90.3**%	3.357

Table 3: Segmentation accuracy and time (by seconds) on KITTI dataset. For each column the best is highlighted. Column *overall* reports the overall segmentation accuracy, and column *mean* shows the accuracy as an average of accuracies of different classes. Among the methods that estimate graphs and labels simultaneously, our PDMP method is much faster than Lan.

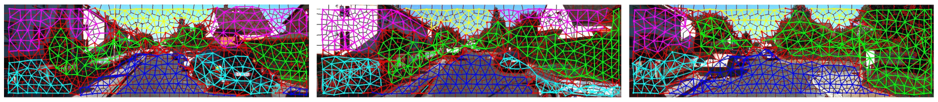
to estimate graph structures, the $-\log$ distance is used rather than Potts. This distance allows to filter highly impossible edges out, *e.g.* the ones with super-pixels far away from each other and distinct in colour. Though $-\log$ feature can be used by other methods, the results are worse than using the exp potentials.

Graph construction. CRF-MST and CRF-Adj use pre-constructed graphs. CRF-MST uses MST computed based on weights that equal the sum of two values: 1) ℓ_2 norm of the difference between the 2D locations of two super-pixels; 2) ℓ_2 norm of the difference between the LAB colour vectors of two super-pixels. CRF-Adj uses graphs consistent with super-pixel adjacency–if two super-pixels are adjacent, their nodes are connected by an edge. For the other two methods, the graphs are estimated together with labels using different inference methods.

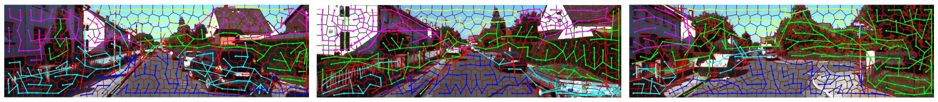
Inference. Since the first two methods use fixed graphs, *i.e.* G is known, belief propagation (BP) can be used to estimate labels. For the last two methods, it is easy to formulate (17) into (1). Hence graphs and labels can be estimated simultaneously using Lan and PDMP approaches respectively. Note using LP or LP+B&B to do inference in this experiment are computationally prohibitive.

Regarding to the model parameter $\mathbf{w}$, the first two approaches use the maximum pseudo likelihood (MPL) method [15] to learn $\mathbf{w}$, while Lan and PDMP use empirically selected $\mathbf{w} = [1, 0.1, 0.2]$. The quantitative results are shown in Table 3. Overall our PDMP method performs much better than all rest methods on accuracy, and is much faster than Lan which estimates graphs as well. Notably, the methods using fixed MRF graphs are much faster than Lan and PDMP since their inference problem is much easier than the problem (1). Visualisation of some segmentation results by different methods is provided in Figure 2. It can be seen that the estimated graphs (e) are more coherent with the layout of objects than tree-structured graphs (c) and adjacency graphs (b). A closer look at the figure suggests that our PDMP algorithm finds less undesirable edges than Lan, *e.g.* the connection between the vegetation and the white box in the right-most column.

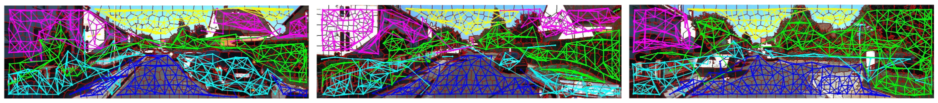
(a) original image



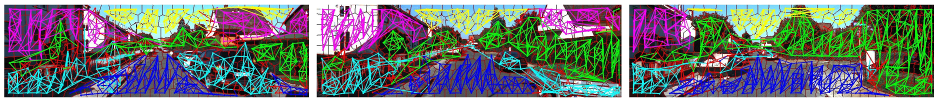
(b) graph according to super-pixel adjacency



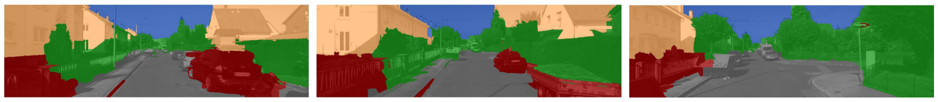
(c) graph obtained via minimum spanning tree



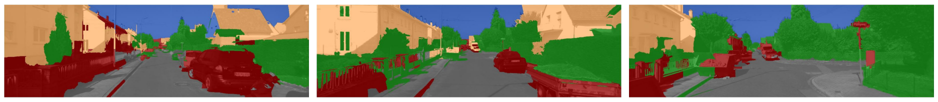
(d) graph estimated via Lan



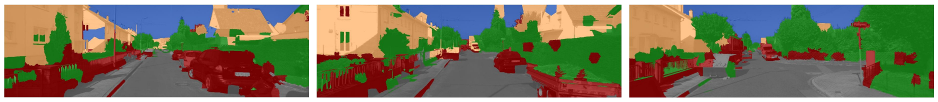
(e) graph estimated via our PDMP algorithm



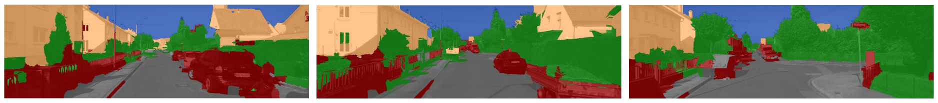
(f) segmentation result by CRF-Adj



(g) segmentation result by CRF-MST



(h) segmentation result by Lan



(i) segmentation result by our PDMP method

Fig. 2: Visualisation of estimated graphs ((b)–(e)) and labels ((f)–(i)) in the image segmentation task. Note a red edge indicates that the label predictions for the associated nodes are different, while edges in other colours indicate identical label predictions (one colour corresponds to one class). Colour code for segmentation results: ⬜ *ground*, 🟧 *building*, 🟩 *vegetation*, 🟥 *objects*, 🟦 *sky*. Our PDMP results are the best in general.

	cross	wait	queue	walk	talk	overall	mean	time
MCSVM	44.1%	47.2%	94.6%	64.9%	94.0%	68.9%	69.0%	**0.001**
SSVM	45.0%	47.2%	95.3%	**65.2**%	96.1%	71.6%	69.8%	0.002
Lan [4]	55.9%	59.7%	94.6%	62.2%	**99.5**%	75.6%	74.4%	0.062
LP [5]	**60.7**%	60.4%	93.6%	47.3%	**99.5**%	75.0%	72.3%	0.044
LP+B&B [5]	55.9%	**61.8**%	**95.7**%	55.4%	**99.5**%	75.4%	73.7%	0.425
PDMP (ours)	59.3%	59.7%	94.6%	60.8%	**99.5**%	**76.2**%	**74.8**%	0.002

Table 4: Results on CAD dataset by different methods. Here *time* means the average running time by seconds. For each column the best is hilighted.

5.2 Human Activity Recognition

We now consider the task of recognising human group activities. For clarity, the term *activity* is used to describe the behaviour of a group of people, while the term *action* refers to the behaviour of an individual. Let $\mathcal{A}$ denote the activity set. Given an image and n body detections, let $\mathbf{x}_0$ denote the descriptor for the whole image, $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n$ denote descriptors for each of n persons, $\mathbf{y} = [y_1, y_2, \dots, y_n]$ ($y_i \in \mathcal{A}$) represent the corresponding action variables, and $a \in \mathcal{A}$ represent the activity variable for the image. Let $G = (\mathcal{V}, \mathcal{E})$ denote a graph spanning all action variables. The potential function $f_{\mathbf{w}}(\mathbf{x}; a, \mathbf{y}, G)$ (proposed in [4]) is given by

$$f_{\mathbf{w}}(\mathbf{x}; a, \mathbf{y}, G) = \mathbf{w}_0^\top \boldsymbol{\phi}_0(\mathbf{x}_0, a) + \sum\nolimits_i (\mathbf{w}_1^\top \boldsymbol{\phi}_1(\mathbf{x}_i, y_i) + \mathbf{w}_2^\top \boldsymbol{\phi}_2(a, y_i)) + \sum\nolimits_{(j,k)\in\mathcal{E}} \mathbf{w}_3^\top \boldsymbol{\phi}_3(\mathbf{x}_j, \mathbf{x}_k, y_j, y_k, a). \tag{18}$$

Here $\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3$ are image-activity feature, image-action feature, action-activity feature and action-action feature defined in [4] [3], $\mathbf{w}_0, \mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$ are model parameters to be learned during training via latent structured SVM (structured SVM is not applicable since the training problem is non-convex), see [4] for details.

To find the best a, we need to maximise (18) over $a, G, \mathbf{y}$. One can formulate this problem into a form (1) (*c.f.* [4]), which can be solved using our PDMP algorithm or the inference methods described at the beginning of Section 4.

Two additional methods are employed as baselines. The first one is the multi-class SVM (MCSVM): we train a multi-class SVM classifier with linear kernel using HoG descriptor extracted from the minimum bounding box area of all human body detections. The second one is structured SVM (SSVM), for which we train a structured SVM [16] to discriminate activities. The potential function used for this method is a special case of (18) by fixing G as MST computed based on 2D distance between body detections. To related inference problem is solved via BP.

[3] To compute these features, we need low-level image descriptors. All evaluating methods using the potential function (18) use the same descriptors extracted by us.

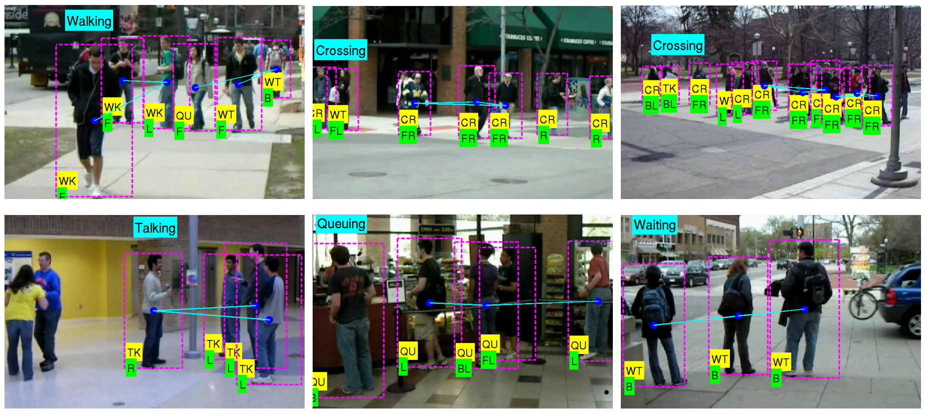


Fig. 3: Visualisation of prediction results on the CAD dataset by PDMP. Activity and action predictions are shown as the texts in cyan and yellow boxes respectively. The human body pose is shown in green boxes. The estimated graph structures are visualised by cyan lines. Abbreviations: cross–CR, walk–WK, wait–WT, Queue–QU, talk–TK, front–F, left-L, right-R, back–B, front-left–FL, front-right–FR, back-left–BL.

We show the results in Table 4. Our PDMP method outperforms all other methods. Please notice using fixed graphs (SSVM) performs much worse than estimating graphs from data (Lan, LP, LP+B&B, PDMP), which verifies the importance of inferring MRF graphs. Comparing with Lan, LP and LP+B&B, our PDMP method performs better because 1) PDMP solves (3) which is tighter than the relaxations used by its competitors; 2) during each iteration PDMP solves a sub-problem of the partial-dual problem (5) exactly. Visualisation of a few recognition results by the PDMP approach is given in Figure 3.

6 Conclusion

We proposed an algorithm to solve MRF inference with unknown graphs. The algorithm is based on the mixed integer bilinear programming formulation, from which we derived its partial-dual and approximately solved the partial dual via message passing. The algorithm scales good to large inference problems without sacrificing performance. We compared our method with existing methods on both synthetic data and real problems. Improvements have been made using our inference technique.

Acknowledgement. We thank Qinfeng Shi for his suggestion on the exposition of this paper. We thank Cesar Dario Cadena Lerma for his help on using the KITTI dataset. This work was supported by a grant from the National High Technology Research and Development Program of China (863 Program) (No. 2013AA10230402), and a grant from the Fundamental Research Funds of Northwest A&F University (No. QN2013056).

References

1. Shotton, J., Winn, J., Rother, C., Criminisi, A.: Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. IJCV **81** (2009) 2–23
2. Pearl, J.: Reverend bayes on inference engines: A distributed hierarchical approach. In: AAAI. (1982) 133–136
3. Nowozin, S., Rother, C., Bagon, S., Sharp, T., Yao, B., Kohli, P.: Decision tree fields. In: ICCV. (2011)
4. Lan, T., Wang, Y., Mori, G.: Beyond actions: Discriminative models for contextual group activities. In: NIPS. (2010)
5. Wang, Z., Shi, Q., Shen, C., van den Hengel, A.: Bilinear programming for human activity recognition with unknown mrf graphs. In: CVPR. (2013)
6. Adams, W.P., Sherali, H.D.: Mixed-integer bilinear programming problems. Mathematical Programming **59** (1993) 279–305
7. Hiroshi, K.: Bilinear programming (1971)
8. Globerson, A., Jaakkola, T.: Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In: NIPS. (2007)
9. Andersen, E., Andersen, K.: Mosek (version 7). Academic version available at www.mosek.com. (2013)
10. Ravikumar, P., Lafferty, J.: Quadratic programming relaxations for metric labeling and markov random field map estimation. In: ICML. (2006)
11. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR. (2012) 3354–3361
12. Sengupta, S., Greveson, E., Shahrokni, A., Torr, P.H.: Urban 3d semantic modelling using stereo vision. In: International Conference on Robotics and Automation. (2013) 580–585
13. Cadena, C., Košecká, J.: Semantic segmentation with heterogeneous sensor coverages. In: International Conference on Robotics and Automation. (2014)
14. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)
15. Koller, D., Friedman, N.: Probabilistic Graphical Models: Principles and Techniques. MIT Press (2009)
16. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR **6** (2006) 1453–1484