Course: Model, Learning, and Inference: Lecture 9

Alan Yuille
Department of Statistics, UCLA
Los Angeles, CA 90095
yuille@stat.ucla.edu

Abstract

Discriminative Random Fields. Grab-Cut. Min-Flow/Max-Cut. Belief Propagation. SWA.
NOTE: NOT FOR DISTRIBUTION!!

1 Introduction

2 Discriminative Random Fields

The purpose of DRFs is to model $P(W|I)$ directly – instead of modeling $P(W)$ and $P(I|W)$.

Let $W = \{w_i\}$ be the set of labels at each image pixel $i \in \Omega$. Write the distribution:

$$P(W|I) = \frac{1}{Z}\exp\{E(W;I)\}. \tag{1}$$

A standard form is:

$$E(W;I) = \sum_{i\in\Omega} E_i(\phi_i(I), w_i) + \sum_{i\in\Omega}\sum_{j\in Nbd(i)} E_{ij}(\phi_{ij}(I), w_i, w_j). \tag{2}$$

This consists of *unary* terms (first terms on RHS) and *binary* terms (second terms on RHS).

Examples: unary terms $E_i(\phi_i(I); w_i) = -\log P(\phi_i(I)|w_i)$. Binary terms $E_{ij}(\phi_{ij}(I) : w_i, w_j) = \delta_{w_i,w_j}\psi(I_i - I_j)$. For example, the unary term gives the probability of the label w_i given the filter response $\phi_i(I)$ (see lecture 1), and the binary term penalizes changes in label between neighbouring points unless there is local evidence such as a big change in image intensity specified by $\phi_{ij}(I)$.

Other examples. See Grab-Cut handout. The goal of Grab-Cut is to start with a rough initial estimate of the boundary of an object (provided by the user) and to produce an accurate segmentation, see figure (3). This task is formalized in terms an energy function with unary and binary potentials, see equation (2), with a binary-valued variable field $\{w_i\}$ with $w_i \in \{0, 1\}$. $w_i = 1$ means that pixel i is inside the object and $w_i = 0$ means that it is outside (i.e., in the background). The unary features are histograms of texture and/or colour filters and are estimated from the initialization – this assumes that the texture statistics are different inside and outside the object (there is contamination because the initialization is not perfect). The binary terms encourage neighbouring pixels to have similar labels unless the intensity values at the pixels is very different. Grab-Cut then uses the Max-Flow/Min-Cut algorithm to estimate $W^* = \arg\min P(W|I)$. This can be performed in stages – estimate W^*, use this to re-estimate the initial statistics of the unary terms, re-estimate W^*, and repeat. (In practice, other terms are added to the energy function encouraging the boundaries of the object to be smooth).

There are several strategies for learning the distributions assuming a training dataset of labelled pixels. The first (sub-optimal) strategy is to learn the unary terms as in lecture 1 and then design a binary term by hand. This is sub-

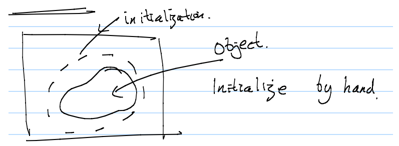

Figure 1: Grabcut uses an initialization of the object boundary to estimate filter statistics inside and outside the object. These are used as unary terms in a discriminative energy function. Then Min-cut/max-flow is used to provide a better estimate of the object boundary.

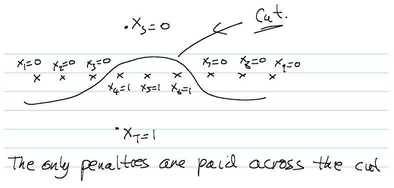

Figure 2: The reformulation of the energy in equation (4) means that the minimum of the energy corresponds to finding the best cut – which divides the pixels into two sets.

optimal since it ignores interactions between the unary and binary terms. The second strategy is to formulate this as a ML problem, which is conceptually straightforward but has all the computational problems associated with ML (difficulty of evaluating the partition function). The third strategy is to apply discriminative learning methods which are alternatives to ML and which will be described later (e.g., structure perceptron and structure max-margin).

3 Energy function and Min-Cut

This is an important class of algorithms for minimizing energy functions which can be guaranteed to converge to the global minimum for some cases. There is a growing literature and new variants of this algorithm are continually being invented.

We now describe the basic idea of how energy minimization can be re-formulated an finding the minimal cut which then relates to finding the max-flow.

Write the energy as a function of binary-valued variables $x_i \in \{0, 1\}$:

$$E(\{x_i\}) = \sum_{ij} a_{ij} x_i x_j + \sum_i a_i x_i + c \tag{3}$$

This can be re-expressed by introducing two new nodes s and t with fixed values $x_s = 0$ and $x_t = 1$:

$$\begin{aligned} E(\{x_i\}) &= -\sum_{ij} a_{ij} x_i (1 - x_j) + \sum_{ij} a_{ij} x_i + \sum_i a_i x_i + c, \\ &= \sum_{ij} a'_{ij} x_i (1 - x_j) + \sum_i a'_i x_i + c, \ \text{ with } a'_{ij} = -a_{ij}, \ a'_i = \sum_j a_{ij} + a_i \\ = \sum_{ij} a'_{ij} x_i (1 - x_j) &+ \sum_{i : a'_i > 0} a'_i x_i (1 - x_s) + \sum_{i : a'_i} |a'_i| (1 - x_i) x_t + c', \ \text{with } c' = c + \sum_{i : a'_i < 0} a'_i. \end{aligned} \tag{4}$$

The energy is now expressed in terms which are only non-zero if the neighbouring nodes take different values. This defines a *min-cut* problem – split $\{x_i\}$ into two sets $X_0 = \{i : x_i = 0\}$ and $X_1 = \{i : x_i = 1\}$, see figure (2). The only penalties are paid across the cut.

Min-cut problems can be solved by max-flow algorithms. There is a general result in the combinatorial optimization literature stating that we can always find the global minimum – or best cut – of $E(\{x_i\})$ provided $a_{ij} \leq 0 \, \forall i, j$. (Note

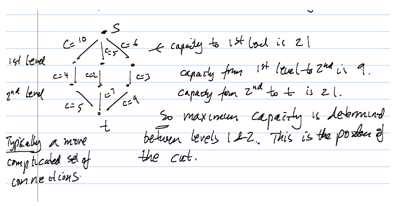
s
c=10
c=5
c=6
← capacity to 1st level is 21
1st level
c=4
c=2
c=3
capacity from 1st level to 2nd is 9.
2nd level
c=5
c=7
c=9
capacity from 2nd to t is 21.
t
So maximum capacity is determined between levels 1 & 2. This is the position of the cut.
Typically a more complicated set of connections.

Figure 3: The maximum flow from source to sink is bounded by the minimum cut determined by the capacities on the edges. Hence finding the maximum flow is equivalent to determining the minimum cut– and hence to minimizing the energy function.

that this differs from the convexity condition for continuous valued functions which states that E has a single global minimum – which hence can be found – if the Hessian of $E(.)$ is positive definite which means that the matrix with terms $\{a_{ij}\}$ is positive definite). More theoretical results can be found in papers by Kolmogorov, Boykov, and others.

Min-cut was originally developed for binary-valued problems. But it can be extended into situations where the x_i can take multiple labels. For example, an α-expansion move is an operation that increases the number of pixels with label α (Kolmogorov and Zabih). Algorithm: find the best α-expansion move. If this has lower energy than current energy then make the move. Repeat for all α. Stop when there is no possible α-expansion. It can be proved that this algorithm will converge to a minimum within a multiplicative factor of the global minimum.

4 Min-Cut/Max-Flow

We need some notation. Graph $G = (V, E)$ where V are the vertices and E is the edges. The edges $(u, v) \in E$ have non-negative capacity $c(u, v) \geq 0$. Source s and sink t.

A *flow* $f : V \times R \mapsto R$ is required to obey the following constraints.

1. *capacity*: $\forall u, v \in V$, we require $f(u, v) \leq c(u, v)$
2. *skew-symmetry*: $\forall u, v \in V$, we require $f(u, v) = -f(v, u)$
3. *flow conservation*: $\forall u \in V/\{s, t\}$, we require $\sum_{v \in V} f(u, v) = 0$.

The *total value* of the flow f is $|f| = \sum_v f(s, v)$.

A *cut* (S, T) us a partition of nodes V into sets S and $T = V - S$ with $s \in S$ and $t \in T$. The *net flow* across the cut (S, T) is $f(S, T)$. The *capacity* of the cut is $c(S, T)$.

If f is a flow in the network (V, E) then f is a maximum flow if, and only if, $|f| = c(S, T)$ for some cut.

So the value of any flow is bounded above by the capacity of any cut. So *the maximum possible flow is given by the minimum cut*. Which minimizes the energy, see equation (4).

Intuitively – the capacity is the restriction in size of pipes that the water can flow down. The size of the pipes determines the maximum amount of flow, see figure (**??**). The capacities are specified by the re-formulation of the energy in equation (4).

The relationship between Min-Cut and Max-Flow motivates the Ford-Fulkerson strategy.

Initialize with $f(u, v) = 0, \forall u, v \in V$ (i.e., initial flow is zero). At each iteration increase the flow by finding an "augmentation path" – a path from source to sink that has the capacity to take more flow. Running time of Ford-Fulkerson depends on how the augmentation path is chosen – e.g., breadth-first search. (Cormen, Leiserson, Rivest).

5 Belief Propagation

BP is an alternative inference algorithm. It reduces to dynamic programming and is guaranteed to converge if the graph has no closed loops (i.e. is a tree). It is surprisingly effective if the graph has closed loops but there are no useful results about this. There is no guaranteed that the algorithm will converge in general.

Suppose the distribution can be represented in the form:

$$P(\vec{x}) = \frac{1}{Z} \prod_{ij} \psi_{ij}(x_i, x_j) \prod_i \psi_i(x_i) \tag{5}$$

BP uses messages $m_{ij}(x_j)$ – the message from node i to node j when node j is in state x_j.

BP can be formulated in terms of the message update equation:

$$m_{ij}(x_j; t+1) = \sum_{x_i} \psi_{ij}(x_i, x_j)\psi_i(x_i) \prod_{k \neq j} m_{ki}(x_i; t), \tag{6}$$

This is the *sum-product* algorithm (Pearl). By replacing the $\sum$ by $\max$ we obtain the *max-product* algorithm (Gallager).

BP gives estimates of the unary and pairwise marginals:

$$\begin{aligned} b_i(x_i; t) &\propto \psi_i(x_i) \prod_k m_{kj}(x_j; t), \\ b_{ij}(x_i, x_j : t) &\propto \psi_k(x_k)\psi_j(x_j)\psi_{kj}(x_k, x_j) \prod_{r \neq j} m_{rk}(x_k; t) \prod_{l \neq k} m_{lj}(x_j; t). \end{aligned} \tag{7}$$

It can be shown (Yedidia et al) that if BP converges then it converges to an extrema of the Bethe Free energy:

$$F = \sum_{ij} \sum_{x_i, x_j} b_{ij}(x_i, x_j) \log \frac{b_{ij}(x_i, x_j)}{\psi_i(x_i)\psi_{ij}(x_i, x_j)\psi_j(x_j)} - \sum_i (n_i - 1) \sum_{x_i} b_i(x_i) \log \frac{b_i(x_i)}{\psi_i(x_i)}. \tag{8}$$

Here n_i is the number of nodes j connected to i. Note that the Bethe Free energies reduces to the standard free energy by setting $b_{ij}(x_i, x_j) = b_i(x_i)b_j(x_j)$. If there are no loops, the BP converges to the (unique) minimum of the Bethe Free energy.

The Bethe Free energy requires lagrange multipliers to enforce the constraints that $\sum_{x_j} b_{ij}(x_i, x_j) = b_i(x_i)$ and $\sum_{x_i} b_i(x_i) = 1$. These lagrange multipliers can be of form $\sum_{ij} \sum_{x_i} \lambda_{ji}(x_i)\{\sum_{x_l} b_{il}(x_i, x_l) - b_i(x_i)\}$. It is straightforward to relate the λ to the messages $m_{ij}(x_j)$. Then BP is in the dual formulation of the Bethe Free energy.

There is an alternative version of BP which does not use messages. We define a local estimates of the distributions, see figure (4) to be:

$$\begin{aligned} B(x_i, x_{N(i)}; t) &= \frac{1}{Z_i} b_i(x_i; t) \prod_{j \in N(i)} \frac{b_{ij}(x_i, x_j; t)}{b_i(x_i; t)}, \\ B(x_i, x_j, x_{N(i,j)}; t) &= \frac{1}{Z_{ij}} b_{ij}(x_i, x_j; t) \prod_{k \in N(i)/j} \frac{b_{ik}(x_i, x_k; t)}{b_i(x_i; t)} \prod_{l \in N(j)/i} \frac{b_{lj}(x_l, x_j; t)}{b_j(x_j; t)}. \end{aligned} \tag{9}$$

Then we can obtain BP by the update rule:

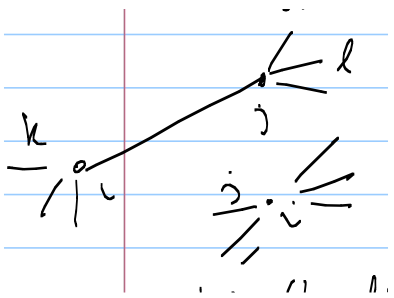

Figure 4: BP without messages corresponds to estimating local probability distributions, from the current estimate of the unary and pairwise beliefs $b_i(x_i, t), b_{ij}(x_i, x_j, t)$. Then the unary and pairwise beliefs are re-estimated from these local probability distributions by marginalizing.

$$b_i(x_i; t+1) = \sum_{x_{N(i)}} B(x_i, x_{N(i)}; t)$$
$$b_{ij}(x_i, x_j; t+1) = \sum_{x_{N(i,j)}} B(x_i, x_j, x_{N(i,j)}; t. \tag{10}$$

Alternative algorithms attempt to directly minimize the Bethe free energy. There are algorithms that can guarantee to minimize the Bethe free energy (Yuille, Kappen and Heskes) but these have two problems: (i) they are double loop algorithms and the convergence proofs breakdown unless the inner loop converges for each iteration (which is hard to be certain of), (ii) sometimes a ”better” solution is obtained by BP even though the double loop algorithms have lower Bethe free energy.

BP can be obtained as a deterministic approximation to Gibbs sampling (Rosen-Zvi, Jordan, Yuille).

Recent re-discovery/re-interpretation of BP by Darwiche and Choi.

There is an important observation that lead to a new class of Tree-Reweighted Algorithms (Wainwright). It follows from the fact that:

$$P(\vec{x}) = \frac{\prod_{ij} \psi_{ij}(x_i, x_j)\psi_i(x_i)\psi_j(x_j)}{\prod_i \{\psi_i(x_i)\}^{n_i - 1}} = \frac{\prod_{ij} b_{ij}(x_i, x_j : t)}{\prod_i \{b_i(x_i : t)\}^{n_i - 1}}. \tag{11}$$

This will be discussed in later lectures.