Notes on Random Fields for Object Recognition

Yaroslav Bulatov

August 6, 2010

Abstract

Suppose you do per-node labeling in graphical model using k-step belief propagation. Can we set parameters in the model to minimize number of labeling errors on training data? Number of errors is hard to minimize, but instead we can minimize a smooth upper bound on the number of errors, like pointwise log-loss

Setting for structured binary labeling

Suppose you are given a set of (x_i, y_i) pairs where $x_i \in \{1,-1\}^m$, $y_i \in \mathcal{F}$ and would like to learn a mapping $\nu : \mathcal{F} \to \cup_m \{1,-1\}^m$ from data to minimize some loss L on this set. More concretely, you learn a function $p : \mathcal{F} \to [0;1]^m$ and label component s of datapoint i as 1 if $p_s(y_i) > 1/2$ and -1 otherwise. The number of classification errors can then be written as follows

$$J = \sum_i \sum_s \text{step}(x_{i,t}(1/2 - p_s(y_i))) \tag{1}$$

Where step(x) returns 1 if x is positive, 0 otherwise. Since this loss is hard to work with, replace it with log-loss as follows

$$J = \sum_i \sum_s \text{logloss}(x_{i,s}, p_s(y_i)) \tag{2}$$

Where logloss is defined as follows

$$\text{logloss}(x,y) = \begin{cases} -\log_2(y) \text{ if } x = 1 \\ -\log_2(1-y) \text{ if } x = -1 \end{cases} \tag{3}$$

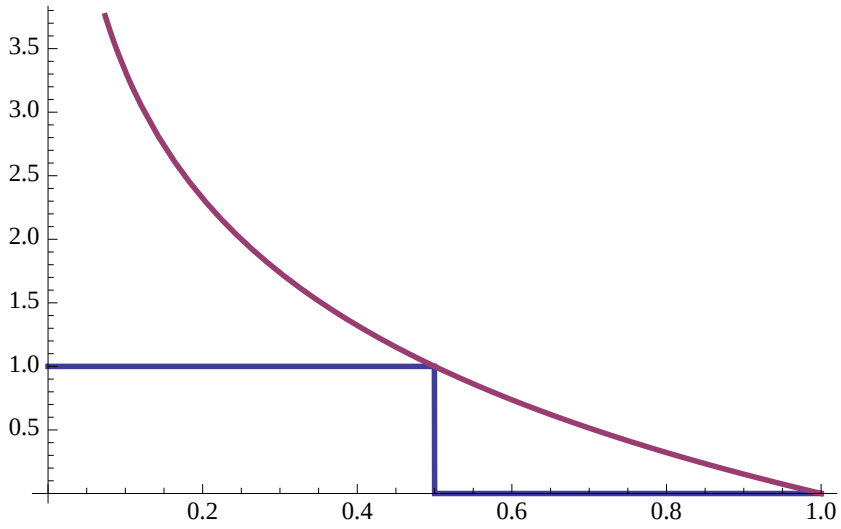

Figure 1: 0-1 loss and log-loss

This loss is smooth, and it forms an upper bound on the 0-1 loss, Figure 1. Minimizing equation 2 minimizes an upper bound on the number of classification errors.

It is more convenient to work with functions that output log-odds instead of probabilities. Define transformation between log-odds and probabilities as follows.

$$o = \frac{1}{2}\log(\frac{p}{1-p}) \qquad p = \frac{\exp(o)}{\exp(o)+\exp(-o)} \tag{4}$$

Let $o(y_i)$ output a vector of log-odds, then the objective function in equation 2 becomes the following

$$J = \sum_i \sum_s \text{logit}(x_{i,s} o_s(y_i)) \tag{5}$$

Where logit (known as logit link function or the logistic loss) is $\text{logit}(x) = \log(1+\exp(-2x))$

Suppose $o(y)$ is defined in terms of parameter vector $\mathbf{w}$. We can minimize 5 by doing gradient descent in space of w, where k'th component of the gradient is defined as

$$\frac{\partial J}{\partial w_k} = \sum_i \sum_s x_{i,s} \frac{\partial \text{logit}}{\partial x} \frac{\partial o_s(y_i)}{\partial w_k} \tag{6}$$

Suppose weights w are parameters defining an undirected graphical model UG and $o_s(y_i)$ represents log-odds of node s being positive obtained by running loopy belief propagation on UG. Next two sections show how to find $\frac{\partial o_s(y_i)}{\partial w_k}$ in this setting.

Random Fields

Suppose $x \in \{1,-1\}^m$. Take graph G with nodes N and edges E, probability density over x's is positive and decomposes according to G. Then it can be written as follows

$$P(x) \propto \exp(\sum_{s\in N} h_s(x_s) + \sum_{st\in E} J_{st}(x_s,x_t)) \tag{7}$$

Where h_s and J_{st} are some potential functions. We can write every density in the form above in the following form

$$P(x) \propto \exp(\sum_{s\in N} \theta_s x_s + \sum_{st\in E} \theta_{st} x_s x_t) \tag{8}$$

Let $C(t)$ indicate the set of children of node t, $C(t)\backslash s$ is children with node s removed. When graph G is a tree following equations define log-odds of the event "x_s is positive".

$$o_s = \sum_{t\in C(s)} m_{st} + \theta_s \tag{9}$$

$$m_{st} = f_{st}(\sum_{u\in C(t)\backslash s} m_{tu} + \theta_t) \tag{10}$$

$$f_{st}(x) = \operatorname{arctanh}(\tanh\theta_{st}\tanh x) \tag{11}$$

Top-down approach to this problem would be to solve this set of equations directly. To define the bottom up approach, let $m \in \mathbb{R}^{2|E|}$ and define the mapping $F : \mathbb{R}^{2|E|} \to \mathbb{R}^{2|E|}$ as follows

$$F_{st}(m) = f_{st}(\sum_{u\in C(t)\backslash s} m_{tu} + \theta_t) \tag{12}$$

Let $F^k = F \circ F \circ \ldots \circ F$ denote k-fold composition of F. Log-odds of x_s can now be written as follows

$$o_s = \sum_{t \in C(s)} F_{st}^k(0) + \theta_s \quad (13)$$

$$k \geq \text{diam}(G) \quad (14)$$

CRF for multi-class object layout

We use the density form from 8 but now θ's are functions of our observed feature variable y.

Let y be a set of bounding boxes with information, where each bounding box has dimensions, detector used to produce that bounding box and score of that detector. Let c_s be the detector responsible for bounding box s (a number between 1 and 20), l_s the score of local detector for box s, r_{st} spatial relationship between boxes s and t (number between 1 and 7, see lines 163-209 in extract_feat_TP.m from multiobject_context package). Then we define potential functions as follows

$$\theta_s(y) = w_{c_s,1} l_s + w_{c_s,0} \quad (15)$$

$$\theta_{st}(y) = w_{c_s,c_t,r_{st}} \quad (16)$$

We now have introduced w's and can finally compute the gradient with respect to w from equation 6. Definition of F is from 12. Let M^k indicate result of belief propagation after k steps, ie

$$M_{st}^k = \sum_{u \in C(t) \setminus s} F_{tu}^k + \theta_t(y_i) \quad (17)$$

Let w_k indicate a local potential offset weight, ie corresponds to $w_{s,0}$ for some s in 16. To find the derivative, apply chain rule to definition of F from 12, f from 11 and θ_s from 16 to get the following

$$\frac{\partial o_s}{\partial w_k} = \frac{\partial \theta_s}{\partial w_k} + \sum_{t \in C(s)} \sum_j \frac{\partial F_{st}^k}{\partial \theta_j} \frac{\partial \theta_j}{\partial w_k} \tag{18}$$

$$\frac{\partial F_{st}^k}{\partial \theta_k} = \frac{\partial f_{st}(M_{st}^k)}{\partial x} \sum_{u \in C(t) \setminus s} \frac{\partial F_{tu}^{k-1}}{\partial \theta_k} + \mathbf{1}(t = k) \tag{19}$$

$$\frac{\partial f_{st}}{\partial x} = \frac{\sinh(2\theta_{st}(y))}{\cosh(2\theta_{st}) + \cosh(x)} \tag{20}$$

$$\frac{\partial \theta_s}{\partial w_k} = \mathbf{1}(c_s = k) \tag{21}$$

Function **I** refers to indicator function, ie 1 if the argument is true, 0 otherwise. Derivatives with respect to other weights $(w_{s,1}, w_{st})$ are derived similarly.

Notice that to you can compute value in equation 20 by using k back-substitutions, so this gives an algorithm similar to k-step belief propagation.

Substitute the value from 18 back into 6 to get the gradient of the objective function.

One step BP and logistic regression

The objective function we minimize in 5 is the same as the objective function minimized for logistic regression. The family of functions we are minimizing over is different, but it becomes the same in a special case.

Suppose our local detector scores l_t are binary valued and we run belief propagation for one step only. Expand definition of o from 14 with $k = 1$ and we get following

$$o_s = \sum_{t \in C(s)} f_{st}(\theta_t(l_t)) + \theta_s(l_s) \tag{22}$$

Since l_t are binary valued, and f_{st}, θ_s have sufficient degrees to freedom to match any function at values 0 and 1, the space of functions we are considering is the same as

$$o_s = \sum_{t \in C(s)} w_{st} l_t + w_s l_s + w_{s,0} \tag{23}$$

This is the same as the family of classifiers we consider when we model log-odds of node t with logistic regression conditioned on detector values at t and neighbours of t

Notation

There's some ambiguity in notation, to resolve consider that s, t, u are always indexing nodes in our graphical model, j, k index components of weight vector w, i indexes elements of our training set. x or x_i couild refers to a labeling (binary valued vector) as in 8, or to just a regular real valued variable 2. Likewise, depending on context, y either means a real valued variable (in 2) or a variable encoding observed feature information.

Many quantities are written as functions without arguments (ie $\frac{\partial \theta}{\partial w}$) but are actually evaluated at particular point, I drop the argument when you can figure it out from context. For instance F^k really means $F^k(0)$ (belief propagation iterated k steps starting with 0 messages). For derivatives I drop the argument when it's the same as argument of original function (ie, f(x) becomes f' after differentiation).

Related work

When marginal likelihoods can be evaluated exactly, objective function of the form 2 for CRF is known as "pointwise log-loss". It's been used before for POS tagging. Also, idea of fitting parameters to make loopy belief propagation produce accurate estimates was suggested by Justin Domke in his PhD thesis, although there he suggests using automatic differentiation tools to compute the gradient.