10-708: Probabilistic Graphical Models 10-708, Spring 2016

24: Max-Margin Learning of GMs

Lecturer: Matthew Gormley *Scribes: Achal Dave, Po-Wei Wang, Eric Wong*

1 Introduction

We've often discussed how there are 5 key ingredients to graphical models: the data, the model, the objective, learning, and inference. This note focuses on the last of these: inference, the task of computing quantities such as likelihoods and marginal probabilities.

1.1 Inference

There are three general tasks for inference.

1. **Marginal Inference**: Compute marginals of variables and cliques:

$$p(\mathbf{x}_C) = \sum_{\mathbf{x}':\mathbf{x}'_C=\mathbf{x}_C} p(\mathbf{x}'|\theta)$$

2. **Partition Function**: Compute the normalization constant:

$$Z(\boldsymbol{\theta}) = \sum_{\mathbf{x}} \prod_{C\in\mathcal{C}} \psi_C(\mathbf{x}_C)$$

3. **MAP Inference**: Compute variable assignment with highest probability.

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} p(\mathbf{x}|\theta)$$

Unfortunately, all three of these are NP-hard in the general case.

Let's focus on the third task, MAP inference. In the past, we've talked about one way to perform MAP inference using Max-Product Belief Propagation, which gives us max-marginals to directly read off the max. In these notes, we'll show that MAP inference can be performed efficiently by solving a linear program.

2 MAP Inference as Mathematical Programming

MAP inference in Chain Markov Net (CRF) Let's assume we are working with acyclic models (a chain model, or a CRF). Recall that a CRF consists of hidden variables $\mathbf{y}$, and observed variables $\mathbf{x}$. The graphical model for a CRF is represented in Figure 1.

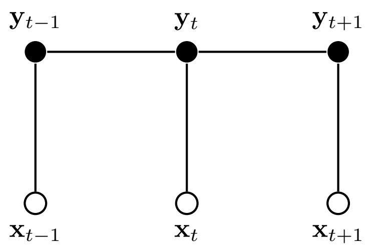

Figure 1: Graphical model for a CRF.

Concretely, we are interested in computing

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \log p_w(y|x) \tag{1}$$

(You may notice that we are using w to represent the model parameters, as opposed to θ, which we used in earlier lectures. This is for consistency with the literature in this area.)

To do MAP inference, we can drop the normalization constant in the distribution.

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} \sum_j \mathbf{w}^T f_{\text{node}}(\mathbf{x}_j, \mathbf{y}_j) + \sum_{j,k} \mathbf{w}^T f_{\text{edge}}(\mathbf{x}_{jk}, \mathbf{y}_j, \mathbf{y}_k) \tag{2}$$

We will show that we can reformulate this as an integer linear program (ILP). We can apply the branch-and-bound method to approximately solve the integer linear program, in which simplex algorithm is applied to solve the linear part. However, if we could relax the integer constraints, we could simply run simplex once and get the exact optimum value for the problem. Fortunately, the following lemma allows us to do exactly that:

Lemma 1 ([Wainwright et al., 2002])**.** *If there is a unique MAP assignment, the linear program relaxation of the integer linear program on a tree structured model is guaranteed to have an integer solution, which is exactly the MAP solution.*

With this lemma in hand, we can now set up the integer program for MAP inference, knowing that we'll be able to solve it efficiently.

We create auxiliary variables z_i corresponding to one hot encodings for each y_i, and auxiliary variables z_{ij} corresponding to one hot encoding matrices for each edge in the matrix.

Concretely, suppose that $y_i \in \{A, B\}$. Then, the sequence $\mathbf{y}$ = 'AB' generates the following auxiliary variables:

$$z_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}^T; \; z_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}^T$$
$$z_{12} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

Now, we can convert the maximization in eq. (2) into a maximization over the encodings z.

$$\begin{aligned}
\max_z \quad & \sum_{j,m} z_j(m)[\mathbf{w}^T f_{\text{node}}(\mathbf{x}_j, m)] + \sum_{j,k,m,n} z_{jk}(m,n)[\mathbf{w}^T f_{\text{edge}}(x_{jk}, m, n)] \\
\text{subject to} \quad & \sum_m z_j(m) = 1 && \text{(normalization)} \\
& \sum_n z_{jk}(m,n) = z_j(m) && \text{(agreement)} \\
& z_j(m) \in \mathcal{Z}, z_{jk}(m) \in \mathcal{Z} && \text{(integer)}
\end{aligned}$$

It turns out that we can rewrite this optimization more compactly as

$$\max_{\mathbf{z}:A\mathbf{z}=\mathbf{b}} (F^T \mathbf{w})^T \mathbf{z}$$

But when our model is acyclic, Lemma 1 shows that we can remove the integer constraints, solve the resulting linear program, and still get an integer(!) solution.

Not covered: dual decomposition. substructures that correspond to linear chains, and other substructures that define a tree, and we have dynamic programming algorithms for each one: dual decomposition allows us to put these together.

3 Max-Margin Markov Networks (M^{3}Ns)

3.1 Motivation 1: Comparison of SVMs and GMs

Handwriting recognition Suppose we have an SVM and we want to distinguish the 'r's from the rest of the characters. The challenge is we have characters that look sort of the same: it is difficult to distinguish between an r and an c, for example. SVMs allow us to use kernels for high dimensional learning and gives generalization bounds.

We would like to use these nice properties of SVMs for learning from sequences. But the number of classes is now exponential in length. With graphical models, the model is instead linear in length. If it were 2002, graphical models don't let us use kernels or generalization bounds, but let us use label correlations. Now, with M^{3}N we can have all three advantages.

Classical Predictive Models Our next motivation is a comparison of loss functions. We have some predictive function, and for learning we want to minimize the loss function with a regularizer. For logistic regression, we have logistic loss and an l_2 regularizer. For SVM we have hinge loss with an l_2 regularizer.

Each of these losses has unique advantages and disadvantages, that are discussed in detail in slide 27 of the lecture. We would like to be able to use these losses in conjunction with graphical models, as opposed to simply performing MAP inference.

The Structured Prediction Problem Classical models described above generally do not incorporate structure amongst labels. Consider, for example, OCR, where we know that the sequence 'brace' is more likely than the sequence 'acrbe'. Classifying each letter individually ignores this structure. Similarly, in dependency parsing of sentences, we have variables between pairs of words and a graphical model that scores edges or triplets.

Structured prediction graphical models Intuitively, M^3N optimizes over the hinge loss for all the sequence, the CRF optimizes over the soft-max loss, and the SVM optimizes over hinge loss but only on binary labels. For M^3N we capitalize on the factorizing structure to solve the problem. Challenges:

1. We typically want an interpretable model. If we have millions of variables and only 10 are non-zero we might actually learn a lot about what these variables are telling us about the data.

2. Prior information of output structures (to be discussed on Wednesday).

3. Latent structures, time series, scalability are other challenges

The main takeaway from M^3Ns is that we'll have strong generalization bounds that will give us better empirical performance.

Parameter Estimation for M^{3}Ns Max conditional likelihood: what we want is to return the answer that gives the maximum possible conditional likelihood, which can be rewritten as $P(y|x) = w^T f(x, y)$. So the dot product of the correct answer should be greater than the dot product of any wrong answer, but there is an exponential number of these. In other words, the goal is to find w such that $w^T(f(x, t(x)) - f(x, y)) > 0$, so $w^T \nabla f_x(y) \geq \gamma \Delta t_x(y)$ and maximize the margin γ.

Estimation: we want to maximize the margin subject to these constraints. Still the problem is an exponential number of constraints on y. Can't just drop into a QP solver, it would be too big to write down.

LP Duality Recap Variables turn into constraints, constraints turn into variables, and the optimal values are the same when both feasible regions are bounded. Specifically, for the following LP

$$\max_z c^T z \quad \text{s.t. } Az \leq b;\ z \geq 0$$

we have that the dual is

$$\min_\lambda b^T \lambda \quad \text{s.t. } A^T \lambda \geq c;\ \lambda \geq 0$$

So in the setting for Markov nets, we change the Markov constraints into a dual variable α, and so we have an exponential number of variables. The constraints is that they sum to 1 and are greater than or equal to 0. These constraints on alpha represent a probability distribution! We can use the insight from graphical models to factorize the summation.

Variable elimination To compute partition functions, slide sums over and use factorization to compute the marginals more efficiently using dynamic programming. We can do this with α as well. Each α is giving a probability over each possible y.

Introduce factored dual variables that is linear in the size of the network: On an acyclic graph, we can factorize αs over nodes and edges with variables μ_i and μ_{ij}. Rewrite the dual using the μ's, then we can formulate the problem as a quadratic program of μ with linear constraints. The first term of the objective is just an expectation of $\Delta t(y)$. The second is just the product of two expectations on $\Delta f(y)$. We can now decompose these expectations over the factorization of α, and rewrite it over the μ.

Factored constraints (when network is a tree; otherwise add clique tree constraint):

$$\begin{cases} \sum_y \alpha(y) = 1 \\ \alpha(y) \geq 0, \forall y \end{cases} \Rightarrow \begin{cases} \sum_i \mu(y_i) = 1, \ \ \sum_{i,j} \mu(y_i, y_j) = 1 & \text{Normalization} \\ \mu(y_i) \geq 0, \ \ \mu(y_i, y_j) \geq 0 & \text{Non-negativity} \\ \mu(y_i) = \sum_j \mu(y_i, y_j) & \text{Agreement} \\ \mu \in \text{CliqueTreePolytope} & \text{Triangulation} \end{cases}$$

Factored dual is now quadratic in network size, and constraint is exponential in tree width. If you wanted to do a tri-gram HMM you could take cliques of larger size than just the edge, still exponential in the tree width.

These constraints are not enough to require a one to one correspondence to a real probability distribution. We have the same set of constraints as in loopy belief propagation, and when these factored dual has not acyclic then the solution will be reasonable but might not correspond to the true α. Has to do with the marginal polytope.

3.2 Min-max formulation

Instead of considering exponentially many constraint in primal

$$\min_w \frac{1}{2}\|w\|, \quad \text{s.t. } w^T f(x, y^*) \geq f(x, y) + \text{loss}(y, y^*), \ \forall y,$$

there is really just one y that has the highest weight, and we just need to consider the most violated constraint instead of every constraint

$$\min_w \frac{1}{2}\|w\|, \quad \text{s.t. } w^T f(x, y^*) \geq \max_{y \neq y^*} \left\{ f(x, y) + \text{loss}(y, y^*) \right\}.$$

Find the 'best' y, and add that one constraint to the problem. Only need a polynomial number of constraints, linear in the training examples. This assumes we have a fixed w, but if we do this during learning, we want to jointly find w as well. You can enforce the constraints one at a time, but here's a different way.

Let's rewrite the constraint as a max within the constraint, and turn the discrete problem into an continuous problem. Rewriting it with indicating one hot encoding variables for the linear chain structure, the problem decomposes into the variables and the edges between variables.

This gives us a transformed problem in terms of the z variables. We can use strong Lagrangian duality to get a third optimization problem with has a nice objective with linear constraints.

Turned exponential constraints to min max problem, replaced with z, switched to dual and jointly optimizing over both sets of variables gives a compact quadratic program.

Experimental Results Handwriting recognition: just trying to predict the right characters given a sequence of characters. Insert results here from the slides. In both cases need to define feature functions over the input images. Over the same feature functions, the M^3 net has lower test error. Use kernel trick to predict over pairs and triples of features.

Hypertext Classification: This graphical model has cycles., we didn't talk about the SMO algorithm, but probably overtaken by exponentiated gradient. See same improvements from the M^3N.

Named entity recognition: Input is a sentence segmented into words, and for each word predict if it's an organization, person, location, or misc. Same improvements

Additional: Proximal / stochastic gradient methods, e.g., MIRA [Crammer et al., 2005], to get around using a quadratic programming solver, and instead use gradient based methods from CRFs.

4 MaxEnDNet

Idea: Using model average and the idea of entropy to learn graphical models. For next time.

References

[Crammer et al., 2005] Crammer, K., McDonald, R., and Pereira, F. (2005). Scalable large-margin online learning for structured classification. In *NIPS Workshop on Learning With Structured Outputs*.

[Wainwright et al., 2002] Wainwright, M., Jaakkola, T., and Willsky, A. (2002). Map estimation via agreement on (hyper) trees: Message-passing and linear programming approaches. In *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, volume 40, pages 1565–1575. The University; 1998.