PGM Course Notes: Approximate Learning Methods

Karl Pichotta
pichotta@cs.utexas.edu

November 21, 2012

1 Gradient Descent

So the learning problem is: given data $x^{(i)}$ drawn iid, estimate θ, the parameters of the exponential family. WE have MLE objective

$$\hat{\theta} = \arg\max_{\theta} \left\{ \theta^T \hat{\mu} - A(\theta) \right\} \equiv \ell(\theta)$$

The solution for $\hat{\mu}$ is

$$\hat{\mu} = \nabla A(\hat{\theta})$$

which is tough in general because we don't usually have the gradient of A.

The Gradient Descent rule is

$$\theta^{t+1} \leftarrow \theta^t + \eta_t \nabla \ell(\theta^t) \tag{1}$$

$$= \theta^t + \eta_t (\hat{\mu} - \nabla A(\theta^t)) \tag{2}$$

So we have to iteratively compute $\mu_t \equiv \nabla A(\theta^t)$, which is basically the inference problem—for example, these are the marginals of the various variables if our sufficient statistics are indicator functions. For large problems, inference is hard—it will take exponential time to compute μ^t. In a sense, we are using inference here as a subroutine during learning.

2 Using Approximate Inference for Approximate Learning

There is nothing stopping us from performing approximate inference and plugging that in:

$$\theta^{t+1} \leftarrow \theta^t + \eta_t (\hat{\mu} - \tilde{\mu}^t)$$

with $\tilde{\mu_t}$ the approximate moments of the graphical model with parameters θ^t that we get from some approximate inference method. Recall, though, that there

are very few guarantees we can make in general about approximate learning methods.

Suppose we have a minimal exponential family. Then

$$\hat{\theta} = (\nabla A)^{-1}(\hat{\mu})$$

because $\hat{\theta}$ satisfies the stationary condition:

$$\hat{\mu} = \nabla A(\hat{\theta})$$

We need the assumption of minimality of the family to guarantee invertibility of ∇A. Recall that

$$\hat{\mu} = \frac{1}{n} \sum_i \phi(x^{(i)}).$$

By the law of large numbers, this converges to the true mean:

$$\hat{\mu} \to_n \mathbb{E}_{P_\theta}[\phi(x)]$$

and, further, we get convergence of $\hat{\theta}$:

$$\hat{\theta} = (\nabla A)^{-1}(\hat{\mu}) \to_n (\nabla A)^{-1}(\mu) = \theta$$

That is,

$$\hat{\theta} \to_n \theta.$$

So the MLE is reasonable: it will converge in the limit to the true distribution.

If we do approximate learning from an arbitrarily large n of data points, though, we will introduce a systematic bias.

Recall the variational principle:

$$A(\theta) = \sum_{\mu \in \mathcal{M}} \{\theta^T \mu - A^\star(\mu)\}$$

Our approximate inference algorithm then will compute something that looks like, e.g.,

$$B(\theta) = \sup_{\mu \in \mathbb{L}} \{\theta^T \mu - A^\star_{Bethe}(\mu)\}$$

So we actually get

$$\tilde{\mu} = \nabla B(\theta)$$

and our gradient descent algorithm will be solving:

$$\tilde{\theta} \equiv \arg\max\{\theta^T \hat{\mu} - B(\theta)\}$$

The gradient is $\hat{\mu} - \nabla B(\theta)$, and so our gradient descent rule is

$$\theta^{t+1} \leftarrow \theta^t + \eta_t(\hat{\mu} - \nabla B(\theta^t))$$

We are using some other function $B(\theta)$ that is an approximation of $A(\theta)$, so we really are solving a different problem and won't in general converge to the true MLE parameters.

What does this converge to? Well, we have the stationary condition

$$\hat{\mu} = \nabla B(\tilde{\theta})$$

so we have

$$\tilde{\theta} = (\nabla B)^{-1}(\hat{\mu})$$

and the law of large numbers gives us the following convergence in the limit:

$$\tilde{\theta} \to_n (\nabla B)^{-1}(\mu)$$

which is certainly not equal to $(\nabla A)^{-1}(\mu)$. That is to say, the solution we get is systematically inconsistent wrt the true answer—we'll converge in the limit to something else.

3 Using Pseudo-likelihoods for Approximate Learning

We don't have a handle on the true moments, and bounding the error of $\tilde{\mu}$ by a constant is NP-hard. What's the way out? We consider a different scheme from the "plug in your favorite approximate inference" paradigm. We have:

$$p(x;\theta) = \exp\{\theta^T \phi(x) - A(\theta)\}$$

The MLE solves

$$\max_\theta \left\{ \frac{1}{n} \sum_i \log p(x^{(i)};\theta) \right\}$$

Consider instead maximizing:

$$\max_\theta \left\{ \frac{1}{n} \sum_{i=1}^n \sum_{s=1}^p \log p(x_s^{(i)} | x_{V\setminus s}^{(i)};\theta) \right\}$$

This is the pseudo-likelihood. We're taking a product of conditional distributions that look like:

$$\tilde{p}(x;\theta) = \prod_{s=1}^p p(x_s | x_{V\setminus s};\theta)$$

Note that this is NOT a proper distribution, but does end up giving us consistent and tractable approximations.

Suppose we have an Ising model

$$p(x;\theta) = \exp\left\{ \sum_s \theta_s x_s + \sum_{s,t} \theta_{st} x_s x_t - A(\theta) \right\}$$

The problem here is the log partition function, which is very difficult to calculate. However, the conditional distribution ends up being the logistic distribution:

$$p(x_s | x_1, x_2, \ldots, x_{s-1}, x_{s+1}, \ldots, x_p) = \frac{\exp\{\theta_s x_s + \sum_{t\in N(s)} \theta_{st} x_s x_t\}}{\exp\{\theta_s + \sum_{t\in N(s)} \theta_{st} x_t\} + 1}$$

That is, given the neighbors of s, all the other variables are effectively constants. So the normalization constant here is tractable—we only need to normalize over the univariate distribution over x_s. So we have an exact form, and we can use our favorite optimization software.

Using the pseudo-likelihoods $\tilde{p}$, as defined above, with these conditional distributions gives us that

$$\hat{\theta} \to_n \theta$$

in the limit. These conditional distributions are typically not great for computing moments, but they do work for computing parameters.

Comparing this to mean field: supposing we have

$$p(x_1, \dots, x_p; \theta)$$

mean-field gives us a q

$$q(x_1, \dots, x_p) = q(x_1)q(x_2) \cdots q(x_p)$$

which is "close" to the original $p(x)$.

Pseudo-likelihood, on the other hand, uses

$$\prod_{s=1}^{p} p(x_s | x_{V \setminus s}; \theta)$$

instead.

It's not introducing a systematic bias. The other approximate methods try to approximate the likelihood, but introduce systematic biases. Pseudo-likelihood makes a poor approximation, but it's non systematically biased.

Can we get a better approximation than pseudo-likelihood? That is, can we get an approximation method that converges faster? It turns out we cannot—pseudo-likelihood converges at the optimal rate. That is to say, it converges at the best possible asymptotic rate we can expect for any algorithm (using information-theoretic notions), assuming noisy observations. That is, the worst-case estimate $\hat{\theta}$

$$\inf_{\hat{\theta}} \sup_{\theta \in \Theta} \|\hat{\theta} - \theta\|_2$$

scales roughly with

$$\sqrt{\frac{p \log p}{n}}$$

which, if we treat p as constant, is $n^{-1/2}$. This rate of convergence is, in fact, achieved with the pseudo-likelihood.

The upshot is that you should probably use pseudo-likelihood for learning.

4 What if we don't know the graph's structure?

Above we considered the graph's structure as fully specified. Consider the Ising model

$$p(x;\theta) = \exp\left\{\sum_s \theta_s x_s + \sum_{s,t} \theta_{st} x_s x_t - A(\theta)\right\} = \exp\{\theta^T \phi(x) - A(\theta)\}$$

So the $\phi(x)$ assumes we have the graph structure—we need to know where the edges are in order to be able to compute the sufficient statistics.

If we know the graph, then we can use learning as we talked about above. If we don't however, what can we do? One option is to use the following basic framework: for $r = 1, \ldots,$ calculate

$$\mathcal{G}^{r+1} \leftarrow someAlgorithm(\mathcal{G}^r)$$

with some sort of greedy algorithm that takes a graph and modifies it at each step. We then, at each r, do gradient descent on the θ's, given $\mathcal{G}^r$.

This is effectively a search algorithm over the space of 2^{p^2} graphs. This is intractable (there are a ton of graphs), and won't in general result in a consistent estimate of θ.

We can, however, use a different method of guessing the graph. We assume it's a complete graph. So for our Ising model, we have

$$p(x;\theta) = \exp\left\{\sum_s \overline{\theta}_s x_s + \sum_{s,t} \overline{\theta}_{st} x_s x_t - A(\overline{\theta})\right\}$$

with

$$\overline{\theta}_s = \theta_s$$

$$\overline{\theta}_{st} = \theta_{st}$$

for $s, t \in E$, and

$$\overline{\theta}_{st} = 0$$

for $s, t \notin E$. So using this notion, we can forget about the graph—if we have a consistent procedure that will converge to the actual parameters, we'll have the convergence $\overline{\theta} \to_n \theta$.

So will we eventually get 0 for the proper $\overline{\theta}$ parameters? Well, we will certainly converge to around 0 eventually. Consider, though,

$$\frac{1}{n^{0.0001}}$$

This is basically a constant 1, even though it eventually converges. So convergence in the limit doesn't necessarily mean we get close to 0 realistically. So we have the following caveats of using pseudo-likelihood to guess $\overline{\theta}$:

1. We'll get a complete graph.

2. The convergence rate will be much slower. This is because we have many more parameters. The dependence on p, the number of parameters, to the convergence rate of the estimates is typically $\sqrt{p}$. That is, the error scales in the root of the number of paramters, so we need many more samples to get good estimates, since $\overline{\theta}$ in general has many more parameters than θ.

We can get around this in a couple ways. First, we can use some sort of two-step iterative procedure, as described at the beginning of this section. Second, we can constrain pseudo-likelihood by using, for example, some sort of regularization, to enforce sparsity.