CS Talks

Computationally feasible greedy algorithms for neural nets February 22, 2016

Lecturer: Andrew Barron Scribe: Kiran Vodrahalli

Contents

1 Overview

Previously, greedy algorithms have been shown to have good approximation and estimation properties for superpositions of a sigmoidal function or other ridge activation functions. In each step the parameters of a new sigmoid are fit to the residuals of the previous sigmoids. However, that has remained a challenging task to produce guarantees for the parameter search for each sigmoid.

Here we discuss developments of two algorithms for this task. One is an implementation of adaptive annealing, in which internal modifications of the sigmoid is made, including parameter squashing and variable replication that permit individual parameter effects to be small, allowing stability of the adaptive annealing solution, at the expense of increasing dimensionality. The other algorithm is a convergent nonlinear modification of the tensor methods of Anandkumar and her colleagues, motivated by optimization of the inner product of the residuals and certain forms of sigmoids, in the context of data from Gaussian or other rotationally invariant distributions. This work is joint with Jason Klusowski.

2 Introduction

My father started a company back in the 60s where they were developoing neural network modeling capabilities, and they continued to do it through the 80s, and I was charged with improving the training routine. I had ad hoc ideas, did this over summers and Christmas breaks. And people would ask "why" did you do that? Tom Cover told me to take as much probability and statistics as I could. Cover's first papers were actually about mutli-layer networks! It shaped my approaches to things from the start. Instead of analyzing a particular fit, my thesis was about what kind of tool could we use for any optimization problem, using information theoretic complexity. The hard part is to capture what the approximations are, what are their descriptions, and how to compute them. Thanks to information theory, the statistical risk is the easy part.

This is the thing that's always been in my mind: How to get bounds on neural nets: Flexible high-dimensional function estimation: You want a flexible model with reasonable performance guarantees that you can compute reasonably well. These bounds have always pointed towards greedy algorithms which can compute them. We're now getting to the place where we can compute them. Now, these are papers which are not finished: These are getting improved day to day. We're going to talk about how if you could do exhaustive search over these parameters, you would naturally expect to get good approximation bounds. We're looking at nonlinear power methods and annealing methods that can give good guarantees.

3 Setup

We have data (X_i, Y_i), $i \in [n]$. We draw $X_i \sim P$, and domain is either a unit cube in $\mathbb{R}^d$ or all of $\mathbb{R}^d$, and the output response variable $Y_i \in \mathbb{R}$. We would like to look at $\mathbf{E}[Y_i|X_i] = f(X_i)$, both in terms of perfect and noisy observations. There's not much difference between these problems, since if you're in high-dimension, you don't have the function everywhere, and the interpolation task is really what dominates things. This also applies to classification problems. We would like to ask which assumptions about f allow us to make headway.

There are lots of building blocks for the types of estimators people use. Activation functions are piecewise constant (sign function), sigmoid, linear spline (i.e. ReLu), sinusoidal, polynomials. These are all single-variable functions, and multivariable functions can be built from them. One of Hilbert's open problems was to characterize whether there are truly multivariable functions: All continuous functions of several variables are compositions of one-dimensional functions, plus some trivial functions (Kolmogorov and L). (Sidenote: A lot of optimization algorithms have these network diagrams).

Tsybenko brought attention to single layer neural net, where we deal with sigmoids, and I began to look at it to in terms of approximation capabilities. Usually, there really are a lot of differents sorts of models which use this sort of framework, where ϕs are products of one-dimensional functions, i.e.: $\phi(x, a) = \phi_1(x_1, a_1) \cdot \cdots \phi_1(x_d, a_d)$. These are product bases. You can also have ridge bases: $\phi(x, a) = \phi(a^T x)$. This talk will focus on ridge bases, there is also need for thinking about product type. Ford-stepwise polynomial regression is of this type, but you don't really know if these succeed. So we want to identify these greedy algorithm situations so we can make some statements.

4 Flexible m-term nonlinear optimization

It is not practical to think about one-shot optimization over all parameters, which is a very high-dimensional optimization problem. From very early on, there have been greedy algorithms. In the 60s, they were trying to use gradient based. They improved upon it at my dad's company by greedy-building of the network. I wanted to study these in the case of a single hidden layer neural net. We built on ideas from Projection-pursuit, and so on. You could just do what is done in statistics, and take the terms you've already chosen, and choose one more term which will make the projection to Y have as small as possible error. A little bit less close to Y is fixing the previous fit, and taking multiples of the new term to get as close as you can to Y. The Greedy algorithm bounds apply to all these cases. I.e. $\hat{f}_m = \alpha \hat{f}_{m-1} + c\phi(x, \theta_m)$ with α_m, c_m, θ_m chosen to achieve $\min_{\alpha,c,\theta} \|Y - \alpha \hat{f}_{m-1} - c\phi_\theta\|^2$. It does not tell you how to do optimization of each term. But actually, there are some bounds $\|f - f_m\| \leq \frac{\|f\|_\theta}{\sqrt{m}}$ and moreover $\|f - f_m\|^2 \leq \inf_g \left(\|f - g\|^2 + \frac{2\|g\|_\phi^2}{2} \right)$.

Now, what is this norm?

$$\|f\|_\phi = \lim_{\epsilon \to 0} \inf \left(\sum_j |c_j| : \| \sum_j c_j \phi_{\theta_j} - f \| \leq \epsilon \right)$$

Basically this is the variation of f with respect to ϕ. This is the atomic norm of ϕ. Considering the case $\|f_\phi\| = 1$, and ϕ closed under sign changes, then the bound follows by induction after you show $\|f - f_m\|^2 \leq (1 - \lambda)^2 \|f - f_{m-1}\|^2 + \lambda^2$. Then picking $\lambda = 1/m$, you can inductively verify that $\|f - f_m\|^2 \leq \frac{1}{m}$.

We would like to also know how this kind of regularity in the neural net is related to other types of representations. L_1 norms in the Fourier domain provide variation bounds of f, whereas L_2 norms are related to norms on the derivatives of the functions. But L_2 norms are a more relaxed assumption on the tail decay of the Fourier transform, and is not enough to give you dimension-independent rates of approximation.

This is the class of functions that reveals that it's critical to do a selection of appropriate terms targeted to that f. So you might think that's cheating, you can't choose terms that are targeted to f: But we can actually get a minimax optimal risk bound via information theory:

$$\mathbf{E}[\|\hat{f}_m - f\|^2] \leq \|f_m - f\|^2 + c\frac{m}{n} \log N(\phi, \delta)$$

where $\log n(\phi, \delta)$ is metric entropy of ϕ at $\delta = 1/m$, and is of the order $d \log(1/\delta)$ and with l_1 constrained parameters, it is of order $(1/\delta) \log d$ thus you can get better results in high-dimensional settings, since you

can get dimension much bigger than sample size: The expected risk is bound by a value not dependent on dimension, though it's a bit worse ($1/n^{1/3}$ instead of $1/n^{1/2}$ in n.

The minimum description length principle leads to complexity penalized least squares criterion with a shared information-theoretic risk bound. The serendipity here is that the right amount of penalty corresponds to description length; it performs as well as if the best number of terms is known in advance. You can also use greedy algorithms to do Lasso and get similar risk bounds.

Now we come to what's new: How are we going to get within a factor of 2 or 10 which would give us the requiste preformance with our greedy algorithms? (Recall that we just stated a greedy procedure, but now how to do this efficiently).

5 New methods

New computational strategies identify approximate maxima with high probability: Third-order Tensor methods, Nonlinear power methods (Barron's term), and general adaptive annealing tools. They are all stochastically initialized search methods. The essential part of the story is that you start randomly and see what happens. Let's talk about tensor methods.

5.1 Tensor methods

Suppose you have a known design distribution $p(X)$, and your target function is $f(x) = \sum_{k=1}^{m} g_k(a_k^T x)$ which is a combination of ridge functions with distinct linearly independent directions a_k. Let's first teach about method of moments. First, relate the moments to parameters in the population. Then invert the relationship to solve for parameters in terms of the moments. We are going to maximize $\mathbf{E}[f(X)\phi(a^T X)]$ or $(1/n)\sum_i Y_i \phi(a^T X_i)$. Then you can do some integration by parts to calculate things like the expected third derivatives of f, even though you only see f by itself. Thus you push differentiate onto $p(X)$, rather than on f itself. We vary that so that we can get $\mathbf{E}[\nabla\nabla^T f(X) g(a^T X)]$. Once we are able to do this, we can use spectral decomposition methods to get the directions to move in. There's something called a **score function** $S^l(X)$ of order l from known $p(X)$. Then, the result is that

$$\mathbf{E}[\frac{\partial^l}{\partial X_{j_i} \partial X_{j_l}}]$$

Well now if f is a linear combination of ridge functions, then the expected Hessian of $f(X)$ gives $M = \sum_{k=1}^{m} a_k a_k^T \mathbf{E}[g_k''(a_k^T X)] = \mathbf{E}[f(X)S^2(X)]$. You could then in principle assume orthogonality (bad assumption) and think of it like spectral decomposition. You can go into third order arrays (see Anandkumar et al 2015, draft) as

$$\sum_{k=1}^{m} a_{j_1,k} a_{j_2,k} a_{j_3,k} \mathbf{E}[g_k'''(a_k^T X)] = \mathbf{E}[f(X) S_{j_1,j_2,j_3}(X)]$$

This is still in progress and some issues exist. There are some issues regarding assumptions about spectral norm, which come from how many terms to use, but I think this can be overcome.

5.2 Scoring Ridge Functions

I want to interpret the task as maximizing an expected product between $f(X)$ and $\phi(a, X)$. So I need to construct the object that will let me do that. We can do that if you come up with a matrix score of a ridge function $M_{a,X} = S^2 g(a^T X) + [S^1 a^T - a(S^1)^T] g'(a^T X) + [aa^T] g'''(a^T X)$.

Then the activation function formed by scoring a ridge function is

$$\phi(a, X) = a^T [M_{a,X}] a = (a^T S^2 a) g(a^T X) + 2(a^T S^1)(a^T a) g'(a^T X) + (a^T a)^2 g''(a^T X)$$

Then, you can interpret

$$\mathbf{E}[f(x)\phi(a, X)] = a^T \mathbf{E}_{\nabla\nabla^T f(X) g(a^T X)}[a]$$

which you see you're taking expectation of a Hessian. Then, $G_k(a_k, a) = \mathbf{E}[g_k''(a_k^T X) g(a^T X)]$ measures the strength of the match of a to direction a_k, which replaces the considerations of Anandkumar where you have to worry about whether the third derivative is zero.

So note that here we are completely dealing with cases where we assume the distribution and see what we can get from there.

5.3 Nonlinear power method

So our goal is to maximize $J(a) = \mathbf{E}[f(X)\phi(a, X)] = a^T M_a a$ s.t. $\|a\| = 1$. Then Cauchy-Schwarz gives $a^T M_a a \leq \|a\|\|M_a a\|$ with equality iff $a \sim M_a a$. This motivates the mapping of the nonlinear power method:

$$V(a) = \frac{M_a a}{\|M_a a\|}$$

We're seeking fixed points $a^* = V(a^*)$ via iterations $a_t = V(a_{t-1})$. If we can verify that $J(a_t)$ is increasing, we can apply our greedy algorithm framework. We can just say we can get within a fraction of the maximum and directly employ our risk bounds, which is a weakness of Anandkumar's approach (risk bounds suffer from having to perturb things right to get details right).

We're making progress on this increasing property, and we've shown it for Hermite polynomials. In fact, using eigen-decompositions, we can re-write $\mathbf{E}[f(X)\phi(a, X)] = a^T M_a a = u^T \tilde{M}_u u$ where

$$\tilde{M}_u = \sum_k \alpha_k \alpha_k^T \tilde{G}_k(\alpha_k, u)\beta/\beta_k$$

and $\tilde{G}_k$ is G_k with a_k and a expressied with α_k and u. The power mapping corresponds to normalizing $u_t = \tilde{M}_{u_{t-1}} u_{t-1}$. This is provably rapidly convergent with $\tilde{G}$ increasing in inner product $\alpha_k^T u$. Then the limit of a_t is a^* proportional to Wu^*.

5.4 Adaptive annealing

We have a partial differential equation we can solve to get the evolution of our estimate of the solution. See

$$\frac{\partial}{\partial t} p_t(\theta) = \nabla^T [G_t(\theta) p_t(\theta)]$$