OPTIMIZATION IN BIG DATA RESEARCH

Course summary

Zixuan Li

student number: 3118103163

professor: Jianyong Sun

December 14, 2018

Contents

1 Research Background

The term "big data" now adays tends to refer to the method of data analytics method rather than the tradional definition. And the machine learning plays a vital role in devising complex models and to predict. In practice Artifical Nerual network is a vital modeling tool of complex data. The most basic component of the neural network is the neuron model. In the neural network, each neuron is connected to other neurons, and when it is stimulated, it will send signal to the other connected neurons. And the connection have different weight, thus has different impact on the other neurons. The total input value received by the neuron will be compared with the threshold value of the neuron and then processed by the activation function to produce the output of the neuron.

Training neural network is to give a training data set, and continuously adjust the transmission weight and activation function in the neural network to reduce the gap between the predicted value and the data set. If the error of the neural network in the training set is expressed by ϵ and it is obviously a function of the connection weight ω and the threshold value θ . This funciton is also called cost function. The training process of the neural network can be regarded as a parameter optimization process. That is, in the parameter space, finding a set of optimal parameters makes ϵ minimum. Therefore optimisation algorithms play a vital role.

This papaer set out to summarize the optimisation methods lectured during the class. Since the author's professional background is not mathematics, there may be mistakes in the summary. The paper will include the following section.

- Gradient descent
- Subgradient descent
- Projected Subgradient descent
- Proximal gradient descent
- Projected proximal gradient descent
- ADMM

2 Gradient descent

The gradient descent method is very easy to understand. The gradient direction indicates the direction where the function grows fastest, so the opposite direction is the direction where the function decreases fastest. For the problem of machine learning model optimization, when we need to solve the minimum value, we can find the optimal value by moving in the direction of gradient descent. We can also interpret gradient descent by quadratic approximation of the function.

$$x^{k+1} = x^k t_k \nabla f(x^k), k = 1, 2, \dots$$

Thus the gradient descent introduced us to the next question, how much step should we take in each iteration. The first option also is the simplist one is to choose fixed step size. If the step is too big, the

algrithm may divergem. And algrithm can be slow if the step is too small. So we need to use convergence analysis which can give us an idea of the right value of step.

Assume that f convex and differentiable, with $dom(f) = \mathbb{R}^n$, and additionally ∇f is Lipschitz continuous with constant L>0, then gradient descent guarantees convergence with convergence rate $O(1/\epsilon)$

Naturely we want to choose the step size adaptively. One way to do this is to use backtracking line search, which limits the descent step with two fix parameter. if the step does not fit the requirement, the step will shrink according to one of the parameter. This method can guarantee convergence. Also, if we want to achieve fasteset convergence speed, we can use Exact line search. The exact line search choose the step that cause the function to descend largest value in each iteration. However, this method is not very efficient as backtracking line search.

Based on grident descent, there are also many optimization algrithm of grident descent. Such as Momentum, Adagrad, RSProp, Adaptive moment estimation (ADAM), etc. See the mlreport2 for more information.

3 Subgradient descent

Subgradient is a generalization of gradient. There can be many subgradients for one function. If the function is differentiable, then the subgradient is the one and only gradient. If not, then the subgradient is in an interval.

A subgradient of a convex function f at x is any $g \in \mathbb{R}^n$

$$f(y) \geq f(x) + g^T(y - x) for \forall y$$

The set of all subgradients is called subdifferential and is a closed convex set. for example: for Indicator Function, subdifferential is a normal cone.

For subgradient we can also perform convex optimization and analysis. So why import the notion of subgradient? Take Lasso problem as an example. Under some conditions, it is nondifferentiable. And the solution to nondifferentiable problems can be achieved through the generalization of gradients – subgradient.

The corresponding optimization condition is

$$f(x^*) = \min_x \Leftrightarrow 0 \in \partial f(x)$$

Eventhough the subgradient optimality do not present an solution for β. We can get a corollary that if $|X_i^T(y - X\beta)| \leq \lambda$ then $\beta_i = 0$.

And when $X = I$ the solution of this simplified lasso problem is the soft-thresholding operator.

Through convergence analysis, subgradient method has convergence rate $O(1/\epsilon^2)$. Comparing with $O(1/\epsilon)$ of gradient descent, it is fairly slow.

And since the sub-gradient do not guarantee the descent, all iterative solutions are retained and then the minimum value is taken.

one tip is that the step size of subgradient descent has to be square summable, not summable.

One example for subgradient descent is Logistic regression.For different step size, there is a big difference between different step of of descent vibration and speed.

Polyak step sizes can automatically get step size and reduces RHS. However the convergence rate won't change, and according to theorem it can not be faster.

Though subgradient has broad applicablility, it has a really slow convergence rate.

4 Projected subgradient descent

Projected subgradient method sets out to optimize a convex set C. The x was project onto C at each iteration and the rest is like the subgradient method.

$$x^k = P_c(x^{k-1} - t_k g^{k-1}), k = 1, 2, \ldots$$

However, the P_c is generally hard to derivate. And the famous alternating projection algorithm is keep projecting between two convex sets.

5 Proximal gradient descent

Proximal gradient methods are a generalized form of projection used to solve non-differentiable convex optimization problems. For decomposable functions:

$$f(x) = g(x) + h(x)$$

f(x):convex and smooth. g(x):convex but not necessarily diffentiable. If f were differentiable, then we can use gradient descent. If $f(x)$ were not differentiable, then we can only make quadratic approximation to $g(x)$.

$$\begin{aligned} x^+ &= \underset{z}{\text{argmin}} \tilde{g}(z) + h(z) \\ &= \underset{z}{\text{argmin}} g(x) + \nabla g(x)^T(x - z) + \frac{1}{2t}\|x - z\|_2^2 + h(z) \\ &= \underset{z}{\text{argmin}} \frac{1}{2t}\|z - (x - t \cdot \nabla g(x))\|_2^2 + h(z) \end{aligned}$$

$\frac{1}{2t}\|z - (x - t \cdot \nabla g(x))\|_2^2$ this can interpret as forcing the z to stay close to the gradient update for g.

And now we can define the proximal mapping

$$\text{prox}_t(x) = \underset{z}{\text{argmin}} \frac{1}{2t}\|x - z\|_2^2 + h(z)$$

Proximal gradient descent can be expressed as:

$$x^{(k)} = \text{prox}_{t_k}\left(x^{(k-1)} - t_k \nabla g\left(x^{(k-1)}\right)\right), k = 1, 2, \cdots$$

similar to gradient descent, we can also use backtracking line search for Proximal gradient descent.

Proximal gradient descent is a generalized gradient descent becasue when minimizing f= g+h. if $h(x) = 0$ it is gradient descent method. if $h = I_c$ it is projected gradient descent. If $g = 0$ it is proximal minimization algorithm.

The projected gradient descent is to use the normal gradient descent update step and then project the step onto C.

And the proximal minimization's update step is

$$x^+ = \underset{z}{\text{argmin}} \frac{1}{2t}\|x - z\|_2^2 + h(z)$$

This method is only implementable if we know prox in closed form.

One exampe of the proximal gradient descent is iterative shrinkage-thresholding algorithm (ISTA). It is an algrithm with $O(1/\epsilon)$ convergence rate. And it can use to solve matrix completion question.

For a lasso problem

$$\min_{\beta \in \mathcal{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 + \lambda\|\beta\|_1$$

consider $g(\beta) = \frac{1}{2}\|y - X\beta\|_2^2, h(\beta) = \|\beta\|_1$ .Then the prox mapping is:

$$\text{prox}_t(\beta) = \underset{z}{\text{argmin}} \frac{1}{2t}\|\beta - z\|_2^2 + \lambda\|z\|_1 = S_{\lambda t}(\beta)$$

$S_\lambda(\beta)$ is the soft-thersholding operator:

$$[S_\lambda(\beta)]_i = \begin{cases} \beta_i - \lambda & \text{if } \beta_i > \lambda \\ 0 & \text{if } -\lambda \leq \beta_i \leq \lambda \\ \beta_i + \lambda & \text{if } \beta_i < -\lambda \end{cases}$$

And The FISTA algorithm is an accelerated ISTA algorithm. Its difference from ISTA is: Nesterove acceleration method is used. However FISTA may not be effective for Matrix Completion.

6 ADMM

This part I haven't understand completely. I only have a gerneral idea of what this method is. I will spend more time understanding this part after the end of the course.

for a convex equality constrained optimization problem, it can be tranformed in to largrangian problem. And it can transformed into a dual problem: maximize $inf_x L(x, y)$. Then the dual ascent update can be

used for the dual problem. The through dual decomposation we can devide the original problem into more small part. And to speed up dual ascent algrithm. The augmented lagrangian which allow the alogrithm to convergen under a more relaxed condition. However it can not be decomposed. And the ADMM alogrithm is employed to solve this question. It combines the decomposability of the dual ascending method with the convergence property of the upper bound of the multiplier method. And ADMM enable us to deploy a master slave architecture solver.

7 Acknowledgements

I would like to express my gratitude to professor Jianyong Sun. Thank you for the profond and vivid lessons you have brought us this semester!