Gradient Descent: Batch, Stochastic, Mini-Batch

Zhang Ruiyang

In Statistical Learning, we often want to find the (global) minimum point of a function, or find the parameter to minimize a function. One way to do this is via **gradient descent**. There are three main kinds of gradient descent: **batch**, **stochastic**, and **mini-batch**. The last one is a mix of the first two.

If we want to optimize a parameter $\mathbf{w}$ with regards to a certain loss function $\mathcal{L}(\mathbf{w})$ of it, we will have the iteration

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla \mathcal{L}(\mathbf{w}^{(t)}),$$

where the η is the learning rate, and we always set it to be positive. We will normally terminate the iteration when the change in parameter after each parameter is sufficiently small.

It is not hard to realise that if there exist a minimum point of $\mathcal{L}(\mathbf{w})$ that is only local but not global, it will have a gradient zero, causing out iteration to terminate. We are, of course, not going to be satisfied with only a local minimum, so what we could do is to repeat the iteration for many times, each with a different initial point $\mathbf{w}^{(0)}$, so that we will likely to not always be stuck at a local minimum point.

1 Batch

The first type of gradient descent is the **Batch Gradient Descent**. This type of gradient descent is going to run over all the data points when we are calculating the $\nabla \mathcal{L}(\mathbf{w})$ at each iteration.

To better understand this, we will take a common example of loss function, the squared error loss

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{D} (y_n - \mathbf{w}^T \mathbf{x}_n)^2$$

where the $\frac{1}{2}$ is an added constant for easy computation, D for the number of data points, and $\mathbf{x}_n$ for the input of each of the data point. At each iteration, we will be calculating the gradient of that function, which is a sum of D terms.

Just by the description, we can see that this is going to be quite computationally expensive, especially when we have a lot of data points. This problem leads us to introduce the following method.

2 Stochastic

The **Stochastic Gradient Descent**, or simply just SGD, is called stochastic since at each iteration, instead of calculating the sum of D data points for the loss function, we will just randomly select one of them, and use that to update of parameter. This, clearly, is much quicker. Another advantage of SGD is that it will less likely to end up on saddle points, since we are only looking at the loss function in one of the D directions.

3 Mini-Batch

If you do not like the SGD and find it not trustworthy enough, you can have a mix of batch and stochastic. This is by, instead of just looking at one random data point for each iteration, looking at several (but certainly much smaller than D) data points at random each time. This will be faster than batch, and more trustworthy than SGD.

–

We can think about the three methods as variations of SGD, where the difference between the three is the number of data points involved in the calculation of each iteration. 1 for SGD, D for batch GD, and a number in between 1 and D for mini-batch GD.

In reality, these methods are not that great as we may potentially expect. If the objective function is not globally convex, we will normally end up at a local minimum, but a relatively good local minimum.

In addition, there is no theoretical support to prove why that is the case, and why we will end up at minimum points, either local or global. That is still an active field of research and the mystery is not completely solved yet.