Abstract

Qunjie Zhou and Zhenzhang Ye

December 3, 2016

Introduction

Artificial neural network

In the last couple of years, deep learning techniques have achieved huge success in various signal and information processing tasks including computer vision, automatic speech recognition and natural language processing, producing/renovating a great deal of state-of-art.

Especially, a variety of deep discriminative models(e.g., deep neural networks, recurrent neural networks, or convolutional neural networks, etc) have been applied to fields of vital importance to modern computer science, and have already brought exiting breakthrough.

Deep learning has such a breathtaking impact, which is not limited on computer science area but on almost every modern technology, that it is worthy of more study and research work.

Training a neural network

It has been widely agreed that the deeper these networks are, the better they can learn to tackle complicated real-world applications. However, deeper networks also result in highly non-convex optimization problems, which requires very good learning/training algorithm.

The traditional back propagation algorithm has the severe problem of being trapped in poor local optima especially when network gets deeper. Recently, stochastic gradient descent has been extremely extensively applied for training of deep neural networks. Although SGD performs fairly well in finding a good approximation of global optima, it is difficult to parallelize across machines, which makes learning at large scale nontrivial. There are now several improved versions(e.g., AdaGrad, Adam, Momentum, etc.), these gradient-based approaches still have the problem of scalability and other issues such as vanishing gradients.

The immediate question comes: Is it the best we can do to efficiently learn a neural network? In fact, viewing training a neural network purely as an optimization problem, gradient descent is surely simple but also naive. In the family of optimization, a bunch of more sophisticated methods such as ADMM, PDHG are very popular as well. Therefore, we believe it's definitely worth trying others to see whether we gain more benefit in either quality or speed.

As our focus is on applying new optimization methods, not exploring deep network architecture, we decide to start simple with a not deep MLP.