Large-Scale Machine Learning with Stochastic Gradient Descent

Léon Bottou

NEC Labs America, Princeton NJ 08542, USA
leon@bottou.org

Abstract. During the last decade, the data sizes have grown faster than the speed of processors. In this context, the capabilities of statistical machine learning methods is limited by the computing time rather than the sample size. A more precise analysis uncovers qualitatively different tradeoffs for the case of small-scale and large-scale learning problems. The large-scale case involves the computational complexity of the underlying optimization algorithm in non-trivial ways. Unlikely optimization algorithms such as stochastic gradient descent show amazing performance for large-scale problems. In particular, second order stochastic gradient and averaged stochastic gradient are asymptotically efficient after a single pass on the training set.

Keywords: stochastic gradient descent, online learning, efficiency

1 Introduction

The computational complexity of learning algorithm becomes the critical limiting factor when one envisions very large datasets. This contribution advocates stochastic gradient algorithms for large scale machine learning problems. The first section describes the stochastic gradient algorithm. The second section presents an analysis that explains why stochastic gradient algorithms are attractive when the data is abundant. The third section discusses the asymptotical efficiency of estimates obtained after a single pass over the training set. The last section presents empirical evidence.

2 Learning with gradient descent

Let us first consider a simple supervised learning setup. Each example z is a pair (x, y) composed of an arbitrary input x and a scalar output y. We consider a *loss function* $\ell(\hat{y}, y)$ that measures the cost of predicting $\hat{y}$ when the actual answer is y, and we choose a family $\mathcal{F}$ of functions $f_w(x)$ parametrized by a weight vector w. We seek the function $f \in \mathcal{F}$ that minimizes the loss $Q(z, w) = \ell(f_w(x), y)$ averaged on the examples. Although we would like to average over the unknown distribution $dP(z)$ that embodies the Laws of

Y. Lechevallier, G. Saporta (eds.), *Proceedings of COMPSTAT'2010*,
DOI 10.1007/978-3-7908-2604-3_16,

Nature, we must often settle for computing the average on a sample $z_1 \ldots z_n$.

$$E(f) = \int \ell(f(x), y)\, dP(z) \qquad E_n(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) \tag{1}$$

The *empirical risk* $E_n(f)$ measures the training set performance. The *expected risk* $E(f)$ measures the generalization performance, that is, the expected performance on future examples. The statistical learning theory (Vapnik and Chervonenkis (1971)) justifies minimizing the empirical risk instead of the expected risk when the chosen family $\mathcal{F}$ is sufficiently restrictive.

2.1 Gradient descent

It has often been proposed (e.g., Rumelhart et al. (1986)) to minimize the empirical risk $E_n(f_w)$ using *gradient descent* (GD). Each iteration updates the weights w on the basis of the gradient of $E_n(f_w)$,

$$w_{t+1} \;=\; w_t - \gamma \frac{1}{n} \sum_{i=1}^{n} \nabla_w Q(z_i, w_t)\,, \tag{2}$$

where γ is an adequately chosen gain. Under sufficient regularity assumptions, when the initial estimate w_0 is close enough to the optimum, and when the gain γ is sufficiently small, this algorithm achieves *linear convergence* (Dennis and Schnabel (1983)), that is, $-\log \rho \sim t$, where ρ represents the residual error.

Much better optimization algorithms can be designed by replacing the scalar gain γ by a positive definite matrix Γ_t that approaches the inverse of the Hessian of the cost at the optimum:

$$w_{t+1} \;=\; w_t - \Gamma_t \frac{1}{n} \sum_{i=1}^{n} \nabla_w Q(z_i, w_t)\,. \tag{3}$$

This *second order gradient descent* (2GD) is a variant of the well known Newton algorithm. Under sufficiently optimistic regularity assumptions, and provided that w_0 is sufficiently close to the optimum, second order gradient descent achieves *quadratic convergence*. When the cost is quadratic and the scaling matrix Γ is exact, the algorithm reaches the optimum after a single iteration. Otherwise, assuming sufficient smoothness, we have $-\log\log \rho \sim t$.

2.2 Stochastic gradient descent

The *stochastic gradient descent* (SGD) algorithm is a drastic simplification. Instead of computing the gradient of $E_n(f_w)$ exactly, each iteration estimates this gradient on the basis of a single randomly picked example z_t:

$$w_{t+1} \;=\; w_t - \gamma_t \nabla_w Q(z_t, w_t)\,. \tag{4}$$