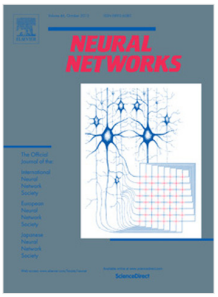
Accepted Manuscript



Accelerating deep neural network training with inconsistent stochastic gradient descent

Linnan Wang, Yi Yang, Renqiang Min, Srimat Chakradhar

PII: S0893-6080(17)30139-9
DOI: http://dx.doi.org/10.1016/j.neunet.2017.06.003
Reference: NN 3768

To appear in: *Neural Networks*

Received date : 13 July 2016
Revised date : 4 May 2017
Accepted date : 4 June 2017

Please cite this article as: Wang, L., Yang, Y., Min, R., Chakradhar, S., Accelerating deep neural network training with inconsistent stochastic gradient descent. *Neural Networks* (2017), http://dx.doi.org/10.1016/j.neunet.2017.06.003

Accelerating Deep Neural Network Training with Inconsistent Stochastic Gradient Descent

Linnan Wang[a,1], Yi Yang[b], Renqiang Min[b], Srimat Chakradhar[b]

[a]*Brown University*
[b]*NEC Laboratories, USA*

Abstract

Stochastic Gradient Descent (SGD) updates Convolutional Neural Network (CNN) with a noisy gradient computed from a random batch, and each batch evenly updates the network once in an epoch. This model applies the same training effort to each batch, but it overlooks the fact that the gradient variance, induced by Sampling Bias and Intrinsic Image Difference, renders different training dynamics on batches. In this paper, we develop a new training strategy for SGD, referred to as Inconsistent Stochastic Gradient Descent (ISGD) to address this problem. The core concept of ISGD is the inconsistent training, which dynamically adjusts the training effort w.r.t the loss. ISGD models the training as a stochastic process that gradually reduces down the mean of batch's loss, and it utilizes a dynamic upper control limit to identify a large loss batch on the fly. ISGD stays on the identified batch to accelerate the training with additional gradient updates, and it also has a constraint to penalize drastic parameter changes. ISGD is straightforward, computationally efficient and without requiring auxiliary memories. A series of empirical evaluations on real world datasets and networks demonstrate the promising performance of inconsistent training.

Keywords: Neural Networks, Stochastic Gradient Descent, Statistical Process Control

1. Introduction

The accessible TFLOPs brought forth by accelerator technologies bolster the booming development in Neural Networks. In particular, large scale neural networks have drastically improved various systems in natural language processing [1], video motion analysis [2], and recommender systems [3]. However, training a large neural network saturated with nonlinearity is notoriously difficult. For example, it takes 10000 CPU cores up to days to complete the training of a network with 1 billion parameters [4]. Such computational challenges have manifested the importance of improving the efficiency of gradient based training algorithm.

The network training is an optimization problem that searches for optimal parameters to approximate the intended function defined over a finite training set. A notable aspect of training is the vast solution hyperspace defined by abundant network parameters. The recent ImageNet contests have seen the parameter size of Convolutional Neural Networks increase to $n \sim 10^9$. Solving an optimization problem at this scale is prohibitive to the second order optimization methods, as the required Hessian matrix, of size $10^9 \times 10^9$, is too large to be tackled by modern computer architectures. Therefore, the first order gradient descent is widely used in training the large scale neural networks.

The standard first order full Gradient Descent (GD), which dates back to [5], calculates the gradient with the whole dataset. Despite the appealing linear convergence rate of full gradient descent ($O(\rho^k)$, $\rho < 1$) [6], the computation in an iteration linearly increases with the size of dataset. This makes the method unsuitable for neural networks trained with the sheer volume of labelled data. To address this issue, Stochastic Gradient Descent [7, 8] was proposed by observing a large amount of redundancy among training examples. It approximates the dataset with a batch of random samples, and uses the stochastic gradient computed from the batch to update the model. Although the convergence rate of SGD, $O(1/\sqrt{bk} + 1/k)$ [9] where b is the batch size, is slower than GD, SGD updates the model much faster than GD in a period, i.e. larger k. As a result, the faster convergence is observable on SGD compared to GD in practice. SGD hits a sweet spot between the good system utilization [10] and the fast gradient updates. Therefore, it soon becomes a popular and effective method to train large scale neural networks.

The key operation in SGD is to draw a random batch from the dataset. It is simple in math, while none-trivial to be implemented on a large-scale dataset such as ImageNet [11]. State of the art engineering approximation is the Fixed Cycle Pseudo Random (FCPR) sampling (defined in section 3.4), which retrieves batches from the pre-permuted dataset like a ring, e.g. $\mathbf{d_0} \rightarrow \mathbf{d_1} \rightarrow \mathbf{d_2} \rightarrow \mathbf{d_0} \rightarrow \mathbf{d_1} \rightarrow \dots$, where $\mathbf{d_i}$ denotes a batch. In this case, each batch receives the same training iterations as a batch updates the network exactly once in an epoch. Please note this engineering simplification allows batches to repetitively flow into the network, which is different from the random sampling in Statistics. However, it is known that the gradient variances differentiate batches in the training [12], and gradient updates from the large loss batch contribute more than the small

Email address: wangnan318@gmail.com (Linnan Wang)