Cost-Sensitive Approach for Batch Size Optimization

Matteo Pirotta
Dept. of Elec., Info. and Bioeng.
Politecnico di Milano, Milan, Italy
matteo.pirotta@polimi.it

Marcello Restelli
Dept. of Elec., Info. and Bioeng.
Politecnico di Milano, Milan, Italy
marcello.restelli@polimi.it

Abstract

In this paper we propose a novel approach to automatically determine the batch size in stochastic gradient descent methods. The choice of the batch size induces a trade-off between the accuracy of the gradient estimate and the cost in terms of samples of each update. We propose to determine the batch size by optimizing the ratio between a lower bound to the first-order Taylor approximation of the expected improvement and the number of samples used to estimate the gradient. The performance of the proposed approach is empirically compared with related methods on an image classification problem using the popular MINST dataset.

1 Introduction

The optimization of the expectation of a function is a relevant problem in large-scale machine learning and in many stochastic optimization problems involving finance, signal processing, neural networks, just to mention a few. The availability of large datasets has called the attention on algorithms that scale favorably both with the number of trainable parameters and the size of the input data. Batch approaches that exploit large samples to compute an approximation of the gradient have been gradually replaced by stochastic approaches that sample a small dataset (usually a single point) per iteration. For example, *stochastic gradient descent* (SGD) methods have been observed to yield faster convergence and (sometimes) less test errors than standard batch methods [1].

In practice, SGD requires several steps of manual adjustment of the parameters to obtain good performance. For example, the initial step size as well as the design of appropriate annealing schema is required for learning with stationary data [2, 3]. In addition, to limit the effects of noisy updates, it is often necessary to exploit mini-batch techniques that require the choice of an additional parameter. These problems are enhanced when nonstationary settings are considered [3].

Several techniques have been designed for the tuning of the step size with pure SGD method [4, 5, 6, 7]. Although these approaches have been successful applied to mini-batch settings, the design of the appropriate *batch size* is still largely unexplored. A notable exception is the work presented in [8] where the authors proposed to adapt the *batch size*, that is the number of samples per gradient update. The batch size is selected according to the variance of the gradient estimated from observed samples. Starting from the geometrical definition of descent direction, through several manipulations, the authors derived the following condition:

$$|S| \geq \frac{\left\| \mathrm{Var}\left[\nabla_{\boldsymbol{\theta}}\right] \right\|_1}{\gamma^2 \left\| \nabla_{\boldsymbol{\theta}}^S \right\|_2^2}, \tag{1}$$

where $\gamma \in (0, 1)$ and $\mathrm{Var}\left[\nabla_{\boldsymbol{\theta}}\right]$ is the vector storing the population variance for each component ($\mathrm{Var}\left[\nabla_{\boldsymbol{\theta}}\right] = [\sigma_1^2, \dots, \sigma_d^2]^\intercal$). The population variance is then approximated through its unbiased estimate $\mathrm{Var}\left[\nabla_{\boldsymbol{\theta}}^S\right]$ computed on sample set S.

The contribute of this paper is the derivation of a novel algorithm for the selection of the batch size in order to compromise between noisy updates and more certain but expensive steps. The proposed

The work was presented at the NIPS workshop on Optimizing the Optimizers. Barcelona, Spain, 2016.

algorithm *automatically* adapts the batch size at each iteration in order to maximize a lower bound to the expected improvement by accounting for the cost of processing samples. The only parameter to be handled is the probability δ that regulates the confidence level of the improvement step.

2 Cost Sensitive Scenario

In this section we formalize the problem and the methodology that will be used in this paper. Consider the problem of maximizing the expected value of a function f

$$\max_{\boldsymbol{\theta}} \mathcal{J}(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}} \mathop{\mathbb{E}}_{x\sim\mathcal{P}} [f(x, \boldsymbol{\theta})],$$

where $\boldsymbol{\theta} \in \mathbb{R}^d$ is the trainable parameter vector and the samples x are drawn i.i.d. from a distribution $\mathcal{P}$. A common approach is to optimize previous function through gradient ascent. However, since $\mathcal{P}$ is unknown it is not possible to compute the *exact* gradient $\nabla_{\boldsymbol{\theta}}\mathcal{J}$ but we can estimate it through samples. Given a training set $S = \{x_i | x_i \sim P, i = 1, \ldots, N\}$, the mini-batch stochastic gradient (SG) ascent is a stochastic process

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \Delta\boldsymbol{\theta}^n \quad = \boldsymbol{\theta}^{(t)} + \boldsymbol{\eta}(t)\nabla_{\boldsymbol{\theta}}\mathcal{J}^n, \qquad t \in \mathbb{N}^+ \tag{2}$$

where $\Delta\boldsymbol{\theta}^n$ is random variable associated to an n-dimensional subset of S (e.g., randomly drawn). Formally $\Delta\boldsymbol{\theta}^n$ is defined as the product of a positive scalar (or positive semi-definite matrix) $\eta(t)$ and a the gradient *estimate* built on a n samples drawn from S

$$\nabla_{\boldsymbol{\theta}}\mathcal{J}^n = \frac{1}{n}\sum_{i\in\mathcal{I}^n} \nabla_{\boldsymbol{\theta}} f(x_i, \boldsymbol{\theta}),$$

where $\mathcal{I}^n$ is a index set used to identify elements in S. $\nabla_{\boldsymbol{\theta}}\mathcal{J}^n$ is a random variable that depends on the selection of the subset of S, i.e., the index set $\mathcal{I}^n$. In the following we will show how to select the batch size n for each gradient update.

To evaluate the quality of an update we consider the improvement $\Delta\mathcal{J}^n = \mathcal{J}(\boldsymbol{\theta} + \Delta\boldsymbol{\theta}^n) - \mathcal{J}(\boldsymbol{\theta})$ that is again a random variable. As the number of samples n increases, the gradient estimate (and consequently the estimated improvement) gets more and more certain. So, we can consider as a goal the maximization of some statistical lower bound Υ^n to the expected improvement $\Delta\mathcal{J}$. On the other hand, this problem is trivially solved by taking the batch size as large as possible, thus not considering the additional computational cost of processing a larger batch size. In practice, this means that the batch size n induces a trade-off between a secure but costly update (the estimate converges to the true value as $n \to +\infty$) and a noisy one. In order to formalize the trade-off, in this paper we consider that any additional sample comes at a price and when the addition of a new sample does not provide any significant improvement in the estimated performance it is not worth to pay that price. As a consequence, we can formalize the batch size selection problem as a *cost sensitive optimization*:

$$n^* = \arg\max_{n\in\mathbb{N}^+} \frac{\Upsilon^n}{n}. \tag{3}$$

3 Lower Bound to the Improvement

This section focuses on the derivation of the lower bound to the improvement $\Delta\mathcal{J}^n$. Given an increment $\Delta\boldsymbol{\theta}^n$, a realization of the random variable $\Delta\mathcal{J}^n$ can be computed. However, we do not know the analytical relationship that ties the two terms. The lack of this information forbids a methodological approach to the problem. For example, we cannot exploit informed methods for the optimization of $\Delta\mathcal{J}^n$ w.r.t. $\Delta\boldsymbol{\theta}^n$, but we can only resort to black-box methods (e.g., grid search).

This relationship can be made explicit by exploiting a Taylor's expansion of the improvement. For example, the first order expansion is given by

$$\Delta\mathcal{J}^n = \nabla_{\boldsymbol{\theta}}\mathcal{J}^{\mathsf{T}}\Delta\boldsymbol{\theta}^n + R_1(\Delta\boldsymbol{\theta}^n), \tag{4}$$

where $R_1(\Delta\boldsymbol{\theta}^n)$ is the remainder. A lower bound to the remainder is easily derived by minimizing the remainder along the line connecting the current parameterization $\boldsymbol{\theta}$ and the value $\boldsymbol{\theta} + \Delta\boldsymbol{\theta}$: $R_1(\Delta\boldsymbol{\theta}) \geq \frac{1}{2}\inf_{c\in(0,1)} \left(\Delta\boldsymbol{\theta}^{\mathsf{T}} H_{\boldsymbol{\theta}}\mathcal{J}(\boldsymbol{\theta} + c\,\Delta\boldsymbol{\theta})\,\Delta\boldsymbol{\theta}\right)$. By plugging in this result in (4), a *deterministic* lower bound to the improvement is derived.

Hoeffding	Chebyshev	Bernstein
$n \geq \frac{18L^2}{\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\|_2^2} \ln\left(\frac{d+1}{\delta}\right)$	$n \geq \frac{9\|\mathrm{Var}[\nabla_{\boldsymbol{\theta}}\mathcal{J}]\|_1}{4\delta\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\|_2^2}$	$n \geq \frac{9b+16a\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\|_2+3\sqrt{9b^2+32ab\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\|_2}}{8\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\|_2^2}$

Table 1: Batch sizes obtained by solving Problem (3) using different concentration inequalities. In the Bernstein case a and b are: $a = \frac{2}{3}L\ln\left(\frac{d+1}{\delta}\right)$ and $b = 2\left\|\mathrm{Var}[\nabla_{\boldsymbol{\theta}}\mathcal{J}]\right\|_2 \ln\left(\frac{d+1}{\delta}\right)$.

The computation of the presented lower bound requires the evaluation of the Hessian in several points (depending on c) which has a quadratic cost in the number of parameters d. The formal way to deal with this issue is to require a high order Lipschitz continuous condition on the objective function in order to derive a bound to the Hessian or to exploit the knowledge of the objective function [9]. However, in practice this information is hard to retrieve and since our goal is to derive a practical algorithm, we suggest to consider the first-order Taylor approximation of the improvement $\Delta\mathcal{J}^n \approx \nabla_{\boldsymbol{\theta}}\mathcal{J}^{\mathsf{T}}\Delta\boldsymbol{\theta}^n$, by dropping the remainder. By adopting a risk-averse perspective, in the next section we describe how to to derive different lower bounds to such approximation of the expect improvement and the correspondent optimal batch sizes obtained by maximizing the ratio between the lower bounds and the number of samples used to estimate the gradient.

4 Linear Probabilistic Adaptive Sample Technique (L-PAST)

Let $\widehat{\Delta}\mathcal{J}^n = \nabla_{\boldsymbol{\theta}}\mathcal{J}^{\mathsf{T}}\Delta\boldsymbol{\theta}^n$ be the linear simplification of the expect improvement. We need to manipulate such formulation in order to remove the dependence on the true gradient. This goal can be achieved by exploiting concentration inequalities on the exact gradient $\nabla_{\boldsymbol{\theta}}\mathcal{J}$. Formally, we consider that the following inequality holds with probability (w.p.) $1-\delta$

$$\left\|\nabla_{\boldsymbol{\theta}}\mathcal{J} - \nabla_{\boldsymbol{\theta}}\mathcal{J}^n\right\|_2 \leq B_\delta^n. \tag{5}$$

Given the previous inequality it is easy to prove the following bound, w.p. $1-\delta$

$$\widehat{\Delta}\mathcal{J}^n = \nabla_{\boldsymbol{\theta}}\mathcal{J}^{\mathsf{T}}\Delta\boldsymbol{\theta}^n = \eta\,\nabla_{\boldsymbol{\theta}}\mathcal{J}^{\mathsf{T}}\,\nabla_{\boldsymbol{\theta}}\mathcal{J}^n \geq \eta\,\left(\left\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\right\|_2 - B^n_{\|\nabla\|}\right)\left\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\right\|_2 = \Upsilon^n, \tag{6}$$

where we have considered the global step size ($\eta \in \mathbb{R}^+$). The lower bound to the expected improvement depends on the batch dimension through the concentration bound ($\left\|\nabla_{\boldsymbol{\theta}}\mathcal{J}^n\right\|_2$ is a realization of the random variable fixed given the current set $\mathcal{I}^n$). In particular, as the number of samples increases, the empirical error (concentration inequality) decreases leading to better estimates of the improvement. Having derived a sample-based bound to the expected improvement, we can solve the cost-sensitive problem (3) for the "optimal" batch dimension n. L-PAST is outlined in Algorithm 1.

4.1 Concentration Inequalities and Batch Dimension

The bound in (6) provides a generic lower bound to the expected improvement that is independent from the concentration inequality used. It is now necessary to provide an explicit formulation in order to solve Problem (3). Several concentration inequalities have been provided in literature, in this paper we consider Hoeffding's, Chebyshev's and Bernstein's inequalities. Chebyshev's inequality has been widely exploited in literature due to its simplicity, it can be applied to any arbitrary distribution (by knowing the variance). On the other side, Hoeffding's and Bernstein's inequalities require a bounded support of the distribution, i.e., the knowledge of the range of the random variables (here $\nabla_{\boldsymbol{\theta}}f(x_i,\boldsymbol{\theta})$). We use the term *distribution aware* to refer to the scenario where the properties of the distribution are known (e.g., variance and range). Although these values can be *estimated* online from the observed value, the results may be unreliable in the event of poor estimates. Empirical version—that directly account for the estimation error—have been presented in literature [10, 11, 12].

Algorithm 1 L-PAST

Inputs: initial batch size n, sample set S
for t=1 **to** T **do**
  $\mathcal{I}^n \leftarrow \{i | i \in 1,\dots,N \wedge |\mathcal{I}^n| = n\}$
  $\nabla_{\boldsymbol{\theta}}\mathcal{J}^n \leftarrow \frac{1}{n}\sum_{i\in\mathcal{I}^n}\nabla_{\boldsymbol{\theta}}f\left(x_i,\boldsymbol{\theta}^{(t)}\right)$
  $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} + \eta\nabla_{\boldsymbol{\theta}}\mathcal{J}^n$ (η through any technique)
  $n \leftarrow \arg\max_{n\in\mathbb{N}^+}\frac{\Upsilon^n}{n}$
end for

The advantage of using these inequalities is that the batch size can be easily computed in *closed form*, see Table 1 for the distribution aware scenario. It is worth to notice that all the proposed approaches

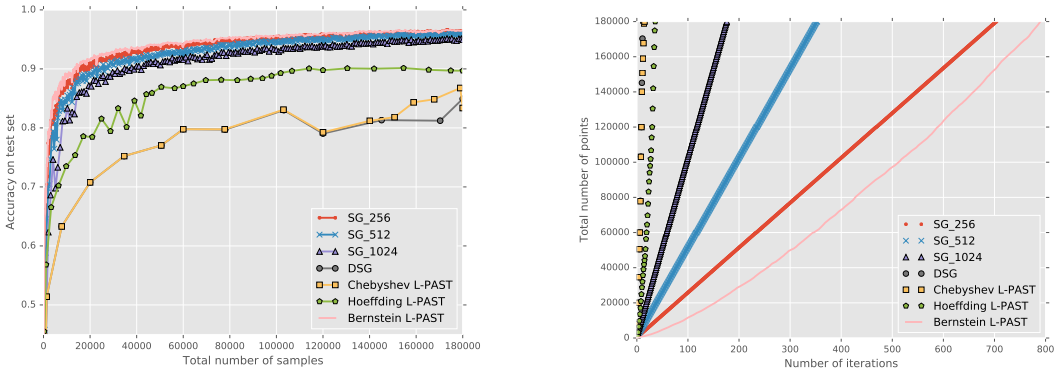

Figure 1: Left figure shows the accuracy on the test set as a function of the total number of samples processed. Right figure draws the total samples w.r.t. the number of iterations. ($\delta = 0.1$)

δ	DSG	H. L-PAST	C. L-PAST	B. L-PAST
0.10	84.93% (13)	89.69% (37)	83.37% (16)	**96.55%** (791)
0.25	87.31% (25)	89.86% (36)	88.33% (27)	**96.64%** (850)
0.50	90.28% (42)	89.25% (43)	91.69% (44)	**96.63%** (902)

Table 2: Accuracy and number of iterations with different confidence levels. SG-256, SG-512 and SG-1024 obtained 96.20% (705), 95.45% (354) and 94.92% (177), respectively.

retains one hyper-parameter $\delta \in (0, 1)$ which denotes the desired confidence level. This parameter can be easily set due to its clear meaning and typically its contribute is small since it is logarithmic.

When we consider the Chebyshev's inequality, i.e., $B^n_\delta = \sqrt{\frac{\|\mathrm{Var}[\nabla_{\boldsymbol{\theta}} \mathcal{J}]\|_1}{n\delta}}$, our approach provides a probabilistic interpretation of the AGSS algorithm presented in [8] and reported in (1). Nevertheless, our derivation gives a different and more formal interpretation of their approach and gives an explicit meaning to the hyper-parameter by mapping $\frac{4\delta}{9}$ to γ^2. It is worth to notice that this result is obtained by considering the distribution aware Chebyshev's inequality instead of the empirical version. By replacing the variance with its estimate the result may be unreliable.

5 Experiments and Conclusions

We choose to test the algorithms on the well known MNIST digit recognition task [13] (with $60k$ training samples and $10k$ test samples). The multi-class classifier is a deep, fully connected multi-layer perceptron with two hidden layers ($[784, 256, 128, 10]$) with activation function RELU. It has *non-convex* loss (cross-entropy) relative to parameters. We tested SG (with batch size of 256, 512 and 1024), DSG [8] and L-PAST with step size selected using RMSprop [14] on 3 epochs. The n-dimensional subset of S is sampled sequentially without shuffling at the beginning of each epoch.

Figure 1 (left) shows that Bernstein L-PAST is able to achieve the best accuracy on the test set. It is possible to observe (right figure) that it selects small batches (in the order of 20-50 samples) in the first iterations when the solution is far from the optimal one, and it increases the batch size when approaching nearly optimal solutions. Other approaches (DSG, Hoeffding/Chebyshev L-PAST) that exploit more general inequalities are prone to select bigger batch sizes that result in less updates and slower convergence rates.

Finally, we tested also different confidence levels. Table 2 shows that, as expected, the confidence δ has a small influence on the overall behavior. It is worth to notice that smaller batches generally lead to better (maybe noisy) performance since are able to perform a larger number of updates.

Pure SG has proved to be effective in several applications, but it is highly time consuming since it exploits one sample for each update. We have shown that it is possible to exploit automatic techniques that are able to adapt the batch size overtime. Moreover, these techniques can be used in conjunction to any schema for the update of the parameters. The L-PAST based on Bernstein's inequality has proved to be effective on the well known MNIST task. Although the batch size may not play a fundamental role in supervised applications, it is a critical parameters in reinforcement learning specially when the environment is highly stochastic (update the estimate with one sample may be too optimistic). Future work will apply the proposed techniques to refine policy gradient approaches.

References

[1] Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Optimization for Machine Learning*, pages 351–368. MIT Press, 2011.

[2] Léon Bottou. *Neural Networks: Tricks of the Trade: Second Edition*, chapter Stochastic Gradient Descent Tricks, pages 421–436. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[3] Tom Schaul, Sixin Zhang, and Yann LeCun. No more pesky learning rates. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Proceedings*, pages 343–351. JMLR.org, 2013.

[4] Abraham P. George and Warren B. Powell. Adaptive stepsizes for recursive estimation with applications in approximate dynamic programming. *Machine Learning*, 65(1):167–198, 2006.

[5] Jan Reinhard Peters. *Machine learning of motor skills for robotics*. PhD thesis, University of Southern California, 2007.

[6] Nicolas Le Roux and Andrew W. Fitzgibbon. A fast natural newton method. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 623–630. Omnipress, 2010.

[7] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[8] Richard H. Byrd, Gillian M. Chin, Jorge Nocedal, and Yuchen Wu. Sample size selection in optimization methods for machine learning. *Math. Program.*, 134(1):127–155, 2012.

[9] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Adaptive step-size for policy gradient methods. In *NIPS 26, December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 1394–1402, 2013.

[10] John G. Saw, Mark C. K. Yang, and Tse Chin Mo. Chebyshev inequality with estimated mean and variance. *The American Statistician*, 38(2):pp. 130–132, 1984.

[11] Volodymyr Mnih, Csaba Szepesvári, and Jean-Yves Audibert. Empirical bernstein stopping. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 672–679, New York, NY, USA, 2008. ACM.

[12] Bartolomeo Stellato, Bart Van Parys, and Paul J. Goulart. Multivariate chebyshev inequality with estimated mean and variance. *The American Statistician*, 2016.

[13] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Technical report, 2012.