Gradient Descent (Batch, Stochastic, Mini-Batch)

Arfat Kamal
dept. Electronic Engineering (ELE)
Hochschule Hamm-Lippstadt (HSHL)
Lippstadt, Germany
arfat.kamal@stud.hshl.de

***Abstract*—Gradient Descent algorithms such as Batch, Stochastic, and Mini-Batch algorithms are used as optimization algorithms in machine learning and they are increasing in popularity over time. These algorithms are often used as black-box optimizers. [1] This article aims to provide an overview of all three Gradient Descent algorithms, their use cases and differences.**

I. Introduction

This paper talks about a machine learning algorithm, Gradient Descent and other variations of it which are Batch, Stochastic and Mini-Batch. These are all machine learning algorithms which are used for different use cases. This paper demonstrates each algorithm, their advantages and disadvantages, and where they are best fitted.

II. Gradient Descent

Gradient descent optimization algorithms, while increasingly popular, are often used as black-box optimizers, as practical explanations of their strengths and weaknesses are hard to come by. This article aims to provide the reader with intuitions with regard to the behaviour of different algorithms that will allow her to put them to use. In the course of this overview, we look at different variants of gradient descent, summarize challenges, introduce the most common optimization algorithms, review architectures in a parallel and distributed setting, and investigate additional strategies for optimizing gradient descent. [1]

Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks. At the same time, every state-of-the-art Deep Learning library contains implementations of various algorithms to optimize gradient descent (e.g. lasagne's, caffe's, and keras' documentation). These algorithms, however, are often used as black-box optimizers, as practical explanations of their strengths and weaknesses are hard to come by.[1]

A. Generalized Nomalized Gradient Descent Algorithm

A generalized normalized gradient descent (GNGD) algorithm for linear finite-impulse response (FIR) adaptive filters is introduced. The GNGD represents an extension of the normalized least mean square (NLMS) algorithm by means of an additional gradient adaptive term in the denominator of the learning rate of NLMS. This way, GNGD adapts its learning rate according to the dynamics of the input signal, with the additional adaptive term compensating for the simplifications in the derivation of NLMS. The performance of GNGD is bounded from below by the performance of the NLMS, whereas it converges in environments where NLMS diverges. The GNGD is shown to be robust to significant variations of initial values of its parameters. Simulations in the prediction setting support the analysis.[2]

B. A General Dynamical Systems Perspective

Learning in biological systems is widely believed to result from progressive modifications occurring at the level of synapses connecting neurons. While the detailed biophysical mechanisms by which a given connection may become stronger or weaker are being intensively investigated,it is clear that ultimately such modifications can depend only on the local electrochemical environment of a given synapse. Obviously, a single synapse does not know anything about the global tasks the organism is trying to learn. Synaptic modifications and global tasks belong to two very distant levels of the brain hierarchy. Therefore, the fundamental question of learning is: What are the principles according to which local synaptic changes are organized in order to yield global learning of complex behaviors for the organism? This puzzling question remains largely unanswered. There are, however, two basic and somewhat complementary ideas in the theoretical literature which have shed some light on this question. [3]

III. Batch Gradient Descent

Vanilla gradient descent, aka batch gradient descent, computes the gradient of the cost function w.r.t. to the parameters.

As we need to calculate the gradients for the whole dataset to perform just one update, batch gradient descent can be very slow and is intractable for datasets that don't fit in memory. Batch gradient descent also doesn't allow us to update our model online, i.e. with new examples on-the-fly.

We then update our parameters in the opposite direction of the gradients with the learning rate determining how big of an update we perform. Batch gradient descent is guaranteed to converge to the global minimum for convex error surfaces and to a local minimum for non-convex surfaces.

IV. Stochastic Gradient Descent

Stochastic gradient descent (SGD) in contrast performs a parameter update for each training example x(i) and label y(i). Batch gradient descent performs redundant computations for large datasets, as it recomputes gradients for similar examples before each parameter update. SGD does away with this

redundancy by performing one update at a time. It is therefore usually much faster and can also be used to learn online. SGD performs frequent updates with a high variance that cause the objective function to fluctuate heavily.

While batch gradient descent converges to the minimum of the basin the parameters are placed in, SGD's fluctuation, on the one hand, enables it to jump to new and potentially better local minima. On the other hand, this ultimately complicates convergence to the exact minimum, as SGD will keep overshooting. However, it has been shown that when we slowly decrease the learning rate, SGD shows the same convergence behaviour as batch gradient descent, almost certainly converging to a local or the global minimum for non-convex and convex optimization respectively. Its code fragment simply adds a loop over the training examples and evaluates the gradient w.r.t. each example. Note that we shuffle the training data at every epoch.

A. Parallelized Stochastic Gradient Descent

With the increase in available data parallel machine learning has become an increasingly pressing problem. In this paper we present the first parallel stochastic gradient descent algorithm including a detailed analysis and experimental evidence. Unlike prior work on parallel optimization algorithms our variant comes with parallel acceleration guarantees and it poses no overly tight latency constraints, which might only be available in the multicore setting. Our analysis introduces a novel proof technique - contractive mappings to quantify the speed of convergence of parameter distributions to their asymptotic limits. As a side effect this answers the question of how quickly stochastic gradient descent algorithms reach the asymptotically normal regime.[4]

B. Stochastic Gradient Boosting

Gradient boosting constructs additive regression models by sequentially fitting a simple parameterized function (base learner) to current "pseudo"-residuals by least squares at each iteration. The pseudo-residuals are the gradient of the loss functional being minimized, with respect to the model values at each training data point evaluated at the current step. It is shown that both the approximation accuracy and execution speed of gradient boosting can be substantially improved by incorporating randomization into the procedure. Specifically, at each iteration a subsample of the training data is drawn at random (without replacement) from the full training data set. This randomly selected subsample is then used in place of the full sample to fit the base learner and compute the model update for the current iteration. This randomized approach also increases robustness against overcapacity of the base learner. [5]

V. Mini-Batch Gradient Descent

Mini-batch gradient descent finally takes the best of both worlds and performs an update for every mini-batch of n training examples.

This way, it a) reduces the variance of the parameter updates, which can lead to more stable convergence; and b) can make use of highly optimized matrix optimizations common to state-of-the-art deep learning libraries that make computing the gradient w.r.t. a mini-batch very efficient. Common mini-batch sizes range between 50 and 256, but can vary for different applications. Mini-batch gradient descent is typically the algorithm of choice when training a neural network and the term SGD usually is employed also when mini-batches are used. Note: In modifications of SGD in the rest of this post, we leave out the parameters for simplicity. [1]

A. Asynchronous Mini-Batch Gradient Descent

With the boom of data, training machine learning model with large-scale datasets becomes a challenging problem. Basing on batch gradient descent (GD) method, researchers propose stochastic gradient descent (SGD) method or mini-batch gradient descent method to relieve the complexity of computation in each iteration and reduce the total time complexity for optimization (Nemirovski et al. 2009; Lan 2012; Ghadimi and Lan 2013; Ghadimi, Lan, and Zhang 2016; Bottou 2010). Due to efficiency, SGD method has been widely used to solve different kinds of large-scale machine learning problems, including both convex and non-convex. However, because we use stochastic gradient to approximate full gradient in the process, a decreasing learning rate has to be applied to guarantee convergence, or it is very easy to diverge from the optimal solution. Thus, it leads to a sub-linear convergence rate of O(1/T) on strongly convex problem. Recently, stochastic variance reduced gradient (SVRG) (Johnson and Zhang 2013) and its variants. [6]

References

[1] Ruder, S., 2022. An overview of gradient descent optimization algorithms. [online] arXiv.org. Available at: ¡https://arxiv.org/abs/1609.04747¿ [Accessed 7 April 2022].

[2] D. P. Mandic, "A generalized normalized gradient descent algorithm," in IEEE Signal Processing Letters, vol. 11, no. 2, pp. 115-118, Feb. 2004, doi: 10.1109/LSP.2003.821649.

[3] P. Baldi, "Gradient descent learning algorithm overview: a general dynamical systems perspective," in IEEE Transactions on Neural Networks, vol. 6, no. 1, pp. 182-195, Jan. 1995, doi: 10.1109/72.363438.

[4] M. Zinkevich, M. Weimer, L. Li, and A. Smola, 'Parallelized Stochastic Gradient Descent', in Advances in Neural Information Processing Systems, 2010, vol. 23. [Online]. Available: https://proceedings.neurips.cc/paper/2010/file/abea47ba24142ed16b7d8fbf2c740e0d-Paper.pdf

[5] Jerome H. Friedman, Stochastic gradient boosting, Computational Statistics Data Analysis, Volume 38, Issue 4, 2002, Pages 367-378, ISSN 0167-9473, https://doi.org/10.1016/S0167-9473(01)00065-2. (https://www.sciencedirect.com/science/article/pii/S0167947301000652)

[6] Huo. (2022). Retrieved 27 May 2022, from https://www.aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14999/14372