Nesterov Accelerated Shuffling Gradient Method for Convex Optimization

The 39th International Conference on Machine Learning (ICML 2022)
July 2022

Trang H. Tran, Katya Scheinberg (Cornell),
Lam M. Nguyen (IBM Research)

Cornell University

IBM Research

INTRODUCTION

Problem Description

We consider the following finite-sum minimization:

$$\min_{w \in \mathbb{R}^d} \left\{ F(w) := \frac{1}{n} \sum_{i=1}^{n} f(w; i) \right\}, \tag{1}$$

where $f(\cdot\,; i) : \mathbb{R}^d \to \mathbb{R}$ is Lipschitz smooth for $i \in [n] := \{1, \ldots, n\}$, and F is convex.

This problem covers many applications in machine learning including logistic regression.

SGD iid vs. Shuffling SGD

1. SGD iid (independently, uniformly distributed)

   - **Uniformly at random:** at each iteration of epoch t, sample an index i uniformly at random from $[n] := \{1, \ldots, n\}$.

1	2	3	...	$n-1$	n

1 epoch $= n$ gradient evaluations

SGD iid vs. Shuffling SGD

1. SGD iid (independently, uniformly distributed)

   - **Uniformly at random:** at each iteration of epoch t, sample an index i uniformly at random from $[n] := \{1, \ldots, n\}$.

1	2	3	...	$n-1$	n

1 epoch $= n$ gradient evaluations

2. Shuffling SGD

   - **Incremental Gradient:** for all epoch t, use a fixed permutation $\pi^{(t)} := \{1, \ldots, n\}$.
   - **Shuffle Once:** at the first epoch $t = 1$, random shuffle a permutation $\pi^{(t)}$ from $[n] := \{1, \ldots, n\}$ and use it for all epochs.
   - **Random Reshuffling:** at epoch t, random shuffle a permutation $\pi^{(t)}$ from $[n] := \{1, \ldots, n\}$.

Nesterov's Accelerated Momentum

Algorithm Nesterov's Accelerated Gradient (NAG)

Initialization: Choose an initial point $x_0, y_0 \in \mathbb{R}^d$.
for $t = 1, 2, \cdots, T$ **do**
  Let $x^{(t)} := y^{(t-1)} - \alpha^{(t)} \nabla F(y^{(t-1)})$
  Compute $y^{(t)} := x^{(t)} + \frac{t-1}{t+2}(x^{(t)} - x^{(t-1)})$
end for

Nesterov's Accelerated Momentum

Algorithm Nesterov's Accelerated Gradient (NAG)

Initialization: Choose an initial point $x_0, y_0 \in \mathbb{R}^d$.
for $t = 1, 2, \cdots, T$ **do**
    Let $x^{(t)} := y^{(t-1)} - \alpha^{(t)} \nabla F(y^{(t-1)})$
    Compute $y^{(t)} := x^{(t)} + \frac{t-1}{t+2}(x^{(t)} - x^{(t-1)})$
end for

- For deterministic convex setting, NAG achieves a much better convergence rate of $\mathcal{O}(1/T^2)$ than that of Gradient Descent $\mathcal{O}(1/T)$ (T is the total number of iterations).
- However, without assuming vanishing variance, the accelerated momentum version of SGD iid do not perform a better theoretical rate than the non-accelerated version.

Nesterov's Accelerated Momentum

- Our goal is using Nesterov's momentum technique for Shuffling SGD to improve the convergence rate.
- The classical approach in stochastic NAG literature is applying the momentum term for each iteration (NASG-PI). However, when inexact gradients are used, NASG-PI might accumulate error [Devolder et al., 2014, Liu and Belkin, 2018].

Nesterov's Accelerated Momentum

- Our goal is using Nesterov's momentum technique for Shuffling SGD to improve the convergence rate.
- The classical approach in stochastic NAG literature is applying the momentum term for each iteration (NASG-PI). However, when inexact gradients are used, NASG-PI might accumulate error [Devolder et al., 2014, Liu and Belkin, 2018].
- We adopt a different approach to update the Nesterov's momentum after each epoch which consists of n gradients.
  This is consistent with the application of Heavy-ball method and proximal operator for shuffling schemes in recent literature [Tran et al., 2021, Mishchenko et al., 2021].

Motivation from experiments

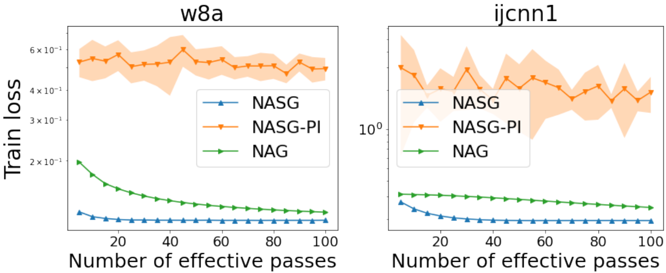

Figure: Comparisons of the training loss for `w8a` and `ijcnn1` datasets.

NAG denotes the deterministic Nesterov's Accelerated Gradient.
While NASG-PI applies Nesterov's momentum each inner iteration, our method NASG applies momentum every epoch.

ALGORITHMS

Nesterov Accelerated Shuffling Gradient

Algorithm Nesterov Accelerated Shuffling Gradient (NASG) Method

Initialization: Choose an initial point $\tilde{x}_0, \tilde{y}_0 \in \mathbb{R}^d$.
for $t = 1, 2, \cdots, T$ **do**
    Set $y_0^{(t)} := \tilde{y}_{t-1}$;
    Generate any permutation $\pi^{(t)}$ of $[n]$
    (either deterministic or random);
    for $i = 1, \cdots, n$ **do**
        Update $y_i^{(t)} := y_{i-1}^{(t)} - \eta_i^{(t)} \nabla f(y_{i-1}^{(t)}; \pi^{(t)}(i))$;
    end for
    Set $\tilde{x}_t := y_n^{(t)}$;
    Update $\tilde{y}_t := \tilde{x}_t + \gamma_t(\tilde{x}_t - \tilde{x}_{t-1})$;
end for

Comparison with deterministic NAG

Algorithm Inner loop of deterministic NAG

1: **for** $i = 1, \cdots, n$ **do**

2: Update $y_i^{(t)} := y_{i-1}^{(t)} - \eta_i^{(t)} \nabla f(y_0^{(t)}; \pi^{(t)}(i))$; $\leftarrow$ fixed point

3: **end for**

Comparison with deterministic NAG

Algorithm Inner loop of deterministic NAG

for $i = 1, \cdots, n$ **do**
  Update $y_i^{(t)} := y_{i-1}^{(t)} - \eta_i^{(t)} \nabla f(y_0^{(t)}; \pi^{(t)}(i))$; $\leftarrow$ fixed point
end for

Algorithm Inner loop of stochastic NASG

for $i = 1, \cdots, n$ **do**
  Update $y_i^{(t)} := y_{i-1}^{(t)} - \eta_i^{(t)} \nabla f(y_{i-1}^{(t)}; \pi^{(t)}(i))$; $\leftarrow$ moving continuously
end for

Main Assumptions

Convexity and standard assumptions

We need the following:

(a) **(Bounded below and convexity for F)** We assume the existence of a minimizer for F, and F is convex.

(b) **(L-smoothness)** $f(\cdot; i)$ is L-smooth for all $i \in [n]$: , i.e., there exists a constant $L > 0$ such that for all $w, w' \in \mathrm{dom}\,(F)$:

$$\|\nabla f(w; i) - \nabla f(w'; i)\| \leq L\|w - w'\|. \tag{2}$$

We let x_* be any minimizer of F and consider the corresponding variance of F at x_*:

$$\sigma_*^2 := \frac{1}{n}\sum_{i=1}^{n} \|\nabla f(x_*; i)\|^2 \in [0, +\infty). \tag{3}$$

Main Assumptions

Additional assumptions

In addition, we assume either (c1) or (c2)

(c1) **(Individual convexity)** $f(\cdot; i)$ is convex for all $i \in [n]$.

(c2) **(Generalized bounded variance)** There exist two finite constants $\Theta, \sigma \geq 0$ such that for any $w \in \mathrm{dom}\,(F)$:

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f(w; i) - \nabla F(w)\|^2 \leq \Theta \|\nabla F(w)\|^2 + \sigma^2. \tag{4}$$

Main results

Theorem 1 (Informal)

We assume Assumption (a) and (b) with either (c1) or (c2) is satisfied. Let $\Delta := \|\tilde{x}_0 - x_*\|^2$ with the initial point $\tilde{x}_0$ and the minimizer x_*. With an appropriate choice of the learning rate, $F(\tilde{x}_T) - F(x_*)$ is upper bounded by

either $\mathcal{O}\left(\frac{\sigma_*^2/L + L\Delta}{T}\right)$, for individual convexity (c1)

or $\mathcal{O}\left(\frac{\sigma^2/(\Theta L) + L\Theta^{1/3}\Delta}{T}\right)$, for generalized bounded variance (c2)

The convergence rate of NASG Algorithm is better than the current state-of-the-art rate [Mishchenko et al., 2020, Nguyen et al., 2021] in term of T for convex problems with general shuffling-type strategies.

Randomized Schemes

Theorem 2 (Informal)

Suppose that Assumption (a), (b) and (c1) hold. Let $\Delta := \|\tilde{x}_0 - x_*\|^2$ with the initial point $\tilde{x}_0$ and the minimizer x_*. With an appropriate choice of the learning rate and randomized shuffling schemes, we have

$$\mathbb{E}[F(\tilde{x}_T) - F(x_*)] \le \mathcal{O}\left(\frac{\sigma_*^2/L}{nT} + \frac{L\Delta}{T}\right),$$

This is better than the corresponding rate for randomized schemes in the literature [Mishchenko et al., 2020, Nguyen et al., 2021] for convex problems.

EXPERIMENTS

Experiments - Logistic Regression

We consider the following convex binary classification problem:

$$\min_{w\in\mathbb{R}^d} \Big\{ F(w) := \frac{1}{n}\sum_{i=1}^{n} \Big[\log(1+\exp(-y_i x_i^\top w)) \Big] \Big\},$$

where $\{(x_i, y_i)\}_{i=1}^n$: a set of training samples, .

We compare our NASG with SGD and two other methods: SGD with Momentum [Polyak, 1964] and Adam [Kingma and Ba, 2014] on three classification datasets `w8a`, `ijcnn1` and `covtype` from `LIBSVM`.

Results - Convex Logistic Regression

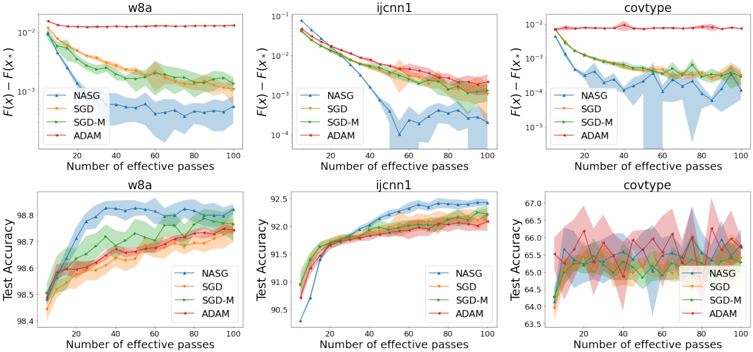

Figure: **(Binary classification).** Comparisons of loss residual $F(x) - F(x_*)$ (top) and test accuracy (bottom) produced by first-order methods for `w8a`, `ijcnn1` and `covtype` datasets, respectively. The number of effective passes is the number of epochs (i.e. number of data passes) in the progress.

Experiments - Neural Networks

We compare our algorithm with SGD and two other methods for image classification problem on MNIST, Fashion-MNIST and CIFAR-10 dataset. We experiment in two settings:

- Convex: linear neural network.
- Non-convex: two-layer neural network.

Figure: Fashion-MNIST dataset (left) and CIFAR-10 dataset (right).[1]

[1] Image source: https://www.tensorflow.org/

Results - Neural Networks

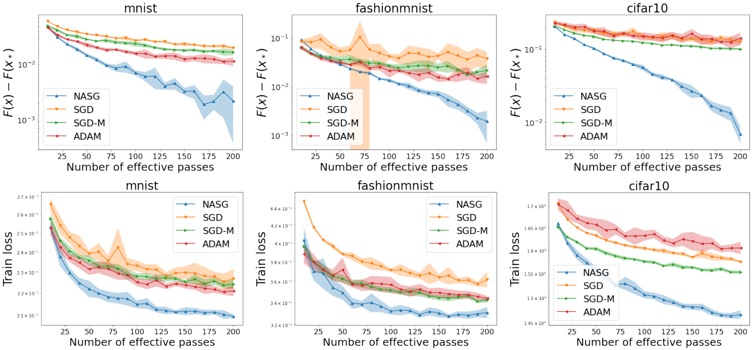

Figure: **(Image classification).** Comparisons of loss residual $F(x) - F(x_*)$ (convex setting, top) and train loss $F(x)$ (non-convex setting, bottom) produced by first-order methods for `MNIST`, `Fashion-MNIST` and `CIFAR-10`, respectively.

Our Contributions

(a) We propose Nesterov Accelerated Shuffling Gradient (NASG) method, which integrates the well-known Nesterov's acceleration technique with shuffling sampling strategies. We adopt a new approach that integrates the momentum for each training epoch.

(b) We establish the convergence analysis for our algorithm in the convex setting using standard assumptions. Our method achieves an improved rate of $\mathcal{O}(1/T)$ in terms of the number of epochs for the unified shuffling schemes. We also investigate the randomized schemes (including Random Reshuffling and Single Shuffling) and improve a factor of n in the convergence bound.

(c) We test our algorithms on various machine learning tasks and compare them with other stochastic first order methods. Our tests have shown good overall performance of the new algorithms.

References I

Devolder, O., Glineur, F., and Nesterov, Y. (2014).
First-order methods of smooth convex optimization with inexact oracle.
Springer-Verlag, 146(1–2).

Kingma, D. P. and Ba, J. (2014).
ADAM: A Method for Stochastic Optimization.
Proceedings of the 3rd International Conference on Learning Representations (ICLR), abs/1412.6980.

Liu, C. and Belkin, M. (2018).
Mass: an accelerated stochastic method for over-parametrized learning.
CoRR, abs/1810.13395.

Mishchenko, K., Khaled, A., and Richtárik, P. (2021).
Proximal and federated random reshuffling.

Mishchenko, K., Khaled Ragab Bayoumi, A., and Richtárik, P. (2020).
Random reshuffling: Simple analysis with vast improvements.
Advances in Neural Information Processing Systems, 33.

References II

Nguyen, L. M., Tran-Dinh, Q., Phan, D. T., Nguyen, P. H., and van Dijk, M. (2021).
A unified convergence analysis for shuffling-type gradient methods.
Journal of Machine Learning Research, 22(207):1–44.

Polyak, B. T. (1964).
Some methods of speeding up the convergence of iteration methods.
USSR Computational Mathematics and Mathematical Physics, 4(5):1–17.

Tran, T. H., Nguyen, L. M., and Tran-Dinh, Q. (2021).
SMG: A shuffling gradient-based method with momentum.
In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10379–10389. PMLR.

THANK YOU!!!

Trang H. Tran - htt27@cornell.edu
https://htt-trangtran.github.io/