# Natasha 2: Faster Non-Convex Optimization Than SGD*

**Zeyuan Allen-Zhu**
zeyuan@csail.mit.edu
Microsoft Research, Redmond

## Abstract

We design a stochastic first-order algorithm to train any smooth neural network to $\varepsilon$-approximate local minima, using $O(\varepsilon^{-3.25})$ backpropagations. The best result was essentially $O(\varepsilon^{-4})$ by SGD.

More broadly, it finds $\varepsilon$-approximate local minima of any smooth nonconvex function in rate $O(\varepsilon^{-3.25})$, with only oracle access to stochastic gradients.

## 1 Introduction

We study the fundamental problem of online stochastic nonconvex optimization:

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) := \frac{1}{n}\sum_{i=1}^n f_i(x) \right\} \tag{1.1}$$

where both $f(\cdot)$ and each $f_i(\cdot)$ can be nonconvex. We want to study

*online algorithms* to find approximate *local minimum* of $f(x)$.

Here, we say an algorithm is online if its complexity is independent of $n$. This tackles the big-data scenarios when $n$ is extremely large or even infinite.[2]

Nonconvex optimization arises prominently in large-scale machine learning (e.g. [7]). Most notably, training *deep neural networks* corresponds to minimizing $f(x)$ of this average structure: each training sample $i$ corresponds to one loss function $f_i(\cdot)$ in the summation. This average structure allows one to perform stochastic gradient descent (SGD) which uses a random $\nabla f_i(x)$ —corresponding to computing backpropagation once— to approximate $\nabla f(x)$ and performs descent updates.

The standard goal of efficient nonconvex optimization is to find local minima, because finding the global one is NP-hard. Experiments in [6, 4, 3] suggest that fast convergence to local minima may be sufficient for training neural nets, while convergence to stationary points (i.e., points that may be saddle points) is *not*. In other words, we need to *escape from saddle points*.

**Escape from Saddle Points.** Randomness naturally helps us escape from saddle points. For instance, Ge et al. [5] showed the random noise incurred by stochastic gradients help SGD make exploitation, and escape from saddle points. Jin et al. [8] showed that, when equipped with random perturbation, full gradient descent (GD) also escapes from saddle points [8]. These results were considered breakthroughs in machine learning, since they successfully explained why stochastic methods perform so well in deep learning. Being easy to implement, however, SGD and GD are still "blind" to the Hessian information of the function.

How can we more effectively use Hessian? Of course, in large-scale settings, we do not wish to apply second-order methods, because even computing a $d \times d$ Hessian matrix may be unrealistic. Fortunately, computing Hessian-vector product is usually computationally cheap.

*Full version available at https://arxiv.org/abs/1708.08694.

[2] All of our results in this paper apply to the case when $n$ is infinite, because we focus on *online* methods. However, we still introduce $n$ to simplify notations.

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

**Our Idea.** Can we escape from saddle points using Hessian-vector products? For instance, instead of naively using a random perturbation vector $v$, can we at least apply power method for a few iterations on $v$, to obtain a better direction in the negative curvature of $\nabla^2 f(x)$?

In this paper, we find a YES answer to this question. In fact, the "correct" *online* variant of power method is known as Oja's algorithm [9]. Earlier this year, Allen-Zhu and Li showed that Oja's algorithm computes the minimum eigenvector of a matrix $\mathbf{M} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{M}_i$ —up to an additive $\delta$ error— using only $\widetilde{O}(\delta^{-2})$ matrix-vector products of the form $\mathbf{M}_i \cdot v$. Therefore, we can use Oja's algorithm to find negative curvature of $\nabla^2 f(x)$, using only Hessian-vector products.[3]

However, finding negative curvature is not enough. Even if $x$ is a point where the Hessian $\nabla^2 f(x)$ has all eigenvalues above some small threshold $-\delta$, its gradient $\nabla f(x)$ may still be large. Thus, we need to find a direction to further decrease $f$, using only first-order information. In other words, can we design an online first-order method that makes use of the fact that $\delta$ is small?

In this paper, we also find a YES to this question. We propose a variance-reduction based online method, that we call `Natasha1.5`, based on the offline method `Natasha1` [1].

Finally, we combine Oja's algorithm and `Natasha1.5` to construct algorithm `Natasha2`, which finds approximate local minima of $f(x)$ using only $T = O(\varepsilon^{-3.25})$ computations of stochastic gradients and Hessian-vector products. Note that $T$ is independent of $n$.

## References


[1] Zeyuan Allen-Zhu. Natasha: Faster Non-Convex Stochastic Optimization via Strongly Non-Convex Parameter. In *ICML*, 2017. Full version available at http://arxiv.org/abs/1702.00763.

[2] Zeyuan Allen-Zhu. Neon2: Finding Local Minima via First-Order Oracles. *ArXiv e-prints*, abs/1711.06673, November 2017. Full version available at http://arxiv.org/abs/1711.06673.

[3] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *AISTATS*, 2015.

[4] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *NIPS*, pages 2933–2941, 2014.

[5] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Proceedings of the 28th Annual Conference on Learning Theory*, COLT 2015, 2015.

[6] I. J. Goodfellow, O. Vinyals, and A. M. Saxe. Qualitatively characterizing neural network optimization problems. *ArXiv e-prints*, December 2014.

[7] Qiuyuan Huang, Paul Smolensky, Xiaodong He, Li Deng, and Dapeng Wu. Tensor product generation networks. *ArXiv e-prints*, abs/1709.09118, 2017.

[8] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to Escape Saddle Points Efficiently. In *ICML*, 2017.

[9] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273, 1982.


[3] In a follow-up work [2], we showed that such Hessian-vector products can be implemented via first-order stochastic gradient computations.