Stochastic Quasi Newton methods for Neural Networks

Konstantin Yakovlev

Optimization Class Project. MIPT

Introduction

The project considers a stochastic quasi-Newtonian method. This project draws heavily on the work of [1]. The algorithm given here uses the classical BFGS update formula in its bounded form. It is also efficient, robust, scalable, and has promising prospects in machine learning. In this project, we propose to implement the above algorithm and compare it with the oBFGS [3] algorithm on the example of logistic regression. In addition, we will compare two optimizers of a neural network – SQGN [2] and SGD.

Problem statement

Let us give some function $f(w, \xi)$, where $w \in \mathbf{R}^n$, ξ is a random vector (x, z). You can look at the pair (x, z) as an object and an answer, respectively. Usually in machine learning, the function $f(w, \xi)$ has the form:

$$f(w, \xi) = \ell(h(w; x_i); z_i),$$

where ℓ is a non-negative loss function, h is a model parameterized by the vector w, $\{x_i, z_i\}_{i=1}^N$ is a training sample. We define the empirical risk as:

$$F(w) = \frac{1}{N}\sum_{i=1}^{N} f(w; x_i, z_i)$$

We formulate the optimization problem as follows:

$$\min_{w \in \mathbf{R}^n} F(w)$$

One can give an estimate of the gradient and the hessian:

$$\widehat{\nabla} F(w) = \frac{1}{b}\sum_{i \in \mathcal{S}} \nabla f(w; x_i, z_i) \qquad (1)$$

$$\widehat{\nabla}^2 F(w) = \frac{1}{b_H}\sum_{i \in \mathcal{S}_H} \nabla^2 f(w; x_i, z_i) \qquad (2)$$

Stochastic quasi-Newton method

Let the optimization method have the following form:

$$w^{k+1} = w^k - \alpha^k H_k \widehat{\nabla} F(w^k) \qquad (3)$$

We introduce the following notation:

$$s_t = \overline{w}_t - \overline{w}_{t-1}, \quad \overline{w}_t = \sum_{i=k-L}^{k} w^i, \quad y_t = \widehat{\nabla}^2 F(\overline{w}_t) s_t$$

Let us define the H_k update:

$$H_{k+1} = (I - \rho_k s_k y_k^\top) H_k (I - \rho_k y_k s_k^\top) + \rho_k s_k s_k^\top, \quad \rho_k = \frac{1}{y_k^\top s_k}$$

Vectors s_t and y_t are calculated every L iterations, where L is a positive integer. In addition, updating the weights on the iteration number $k \leq 2L$ has the following form:

$$w^{k+1} = w^k - \alpha_k \widehat{\nabla} F(w^k)$$

Convergence

Suppose that $F(w)$ is twice continuously differentiable. There exist positive constants λ and Λ such that, for all $w \in \mathbf{R}^n \hookrightarrow \lambda I \prec \nabla^2 F(w) \prec \Lambda$. There is a constant γ such that, for all $w \in \mathbf{R}^n \hookrightarrow E_\xi[\|\nabla f(w^k, \xi)\|_2]^2 \leq \gamma^2$. For convenience, we define $\alpha^k = \frac{\beta}{k}$. Let w^k be the iterates generated by the Newton-like method 3. Where for all k made:

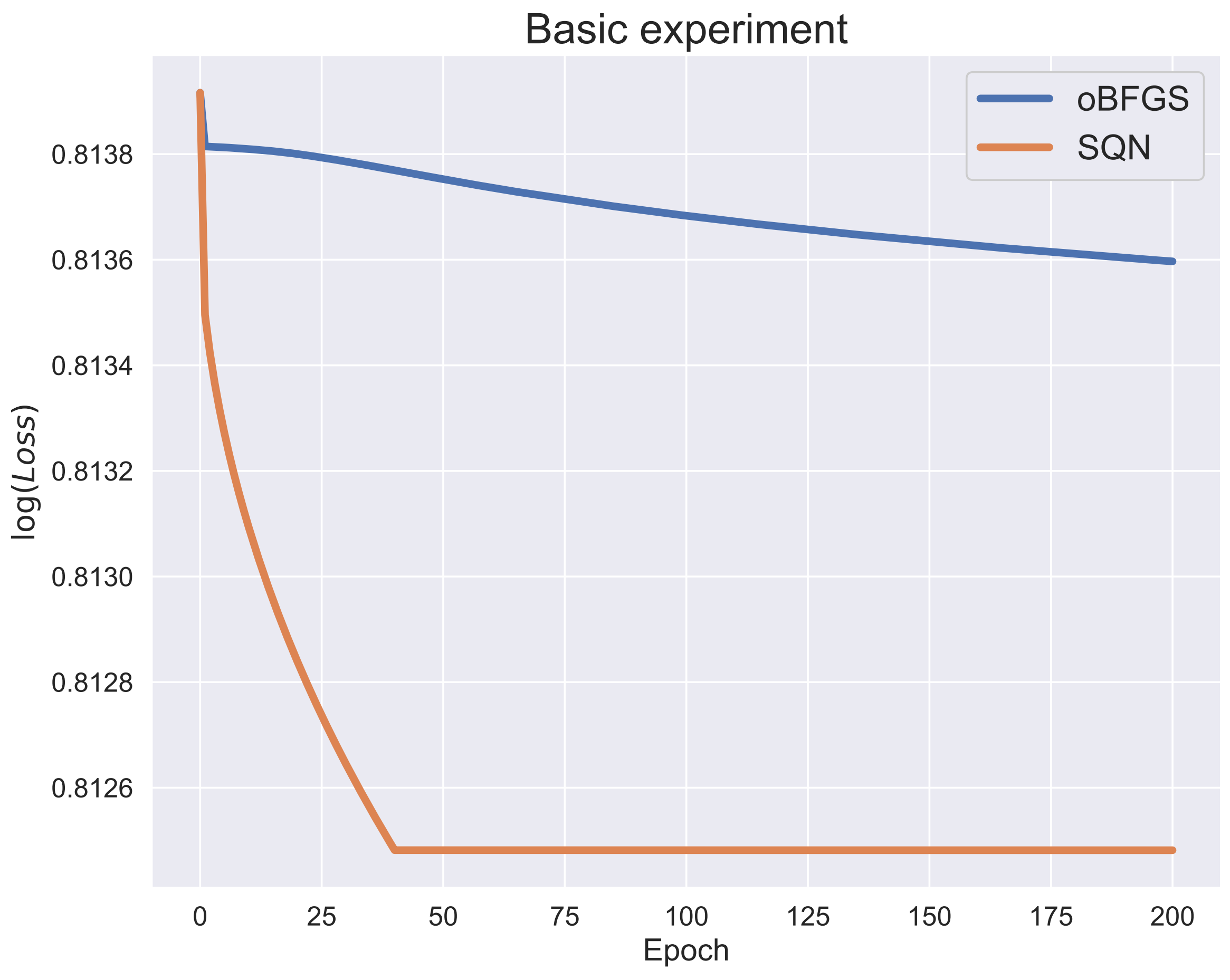
$$\mu_1 I \prec H_k \prec \mu_2 I,\ 0 < \mu_1 \leq \mu_2,\ \beta > (2\mu_1\lambda)^{-1}$$

Then, for all $k \geq 1$

$$E[F(w^k) - F(w_*)] \leq \frac{Q(\beta)}{k},\ Q(\beta) = \max\left(\frac{\Lambda \mu_2^2 \beta^2 \gamma^2}{2(2\mu_1\lambda\beta - 1)}, F(w^1) - F(w_*)\right)$$

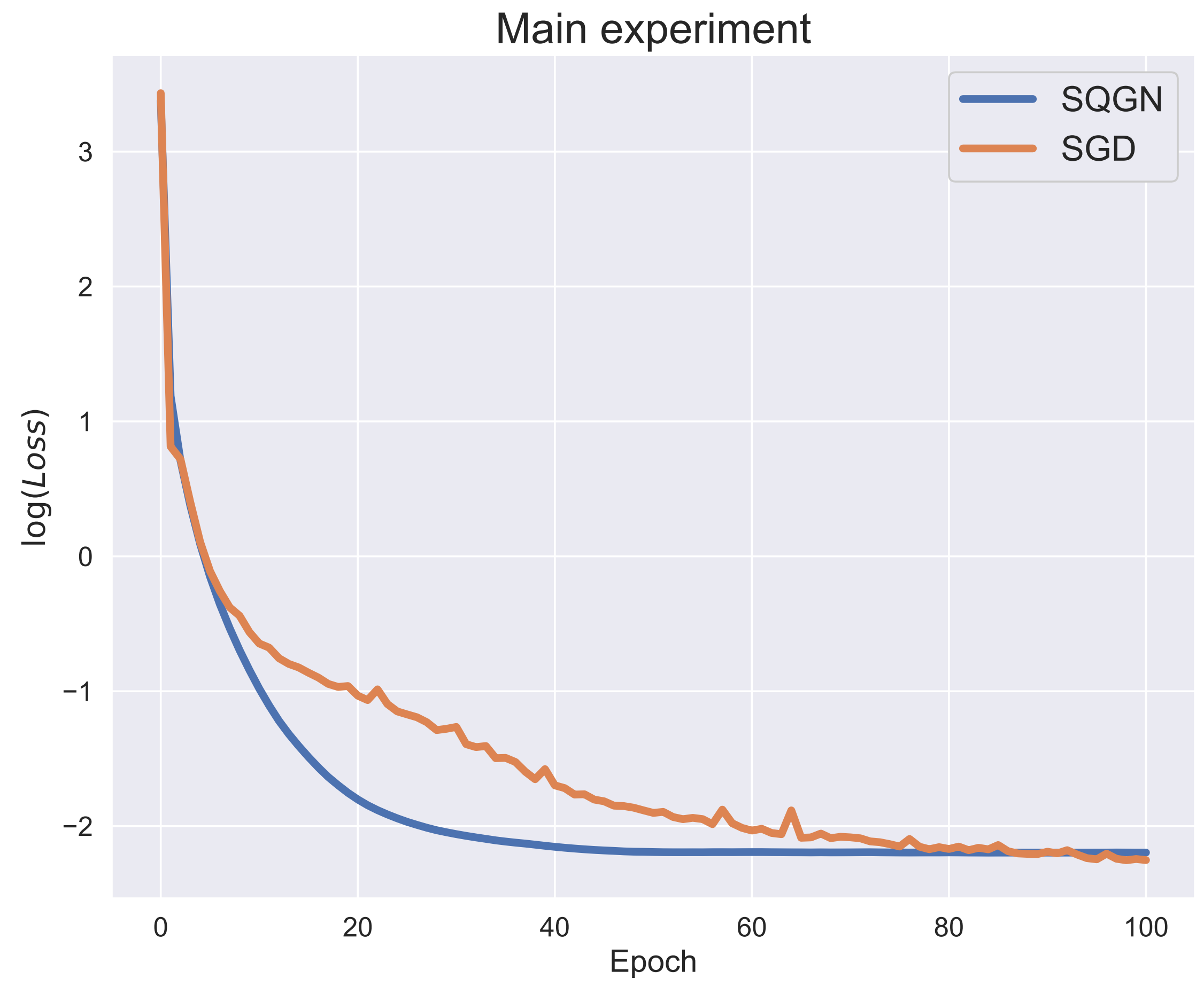
Basic experiment

Here we considered the classification problem, but also the logistic regression. Two optimizers were compared: oLBFGS [3] and SQN [1] on a digits dataset from sklearn library.



Main experiment

Here we compared two optimizers: SQGN [2] and SGD. We considered a TensorFlow-based neural network on the MNIST dataset.



Results

Table 1: Result of the main experiment.

Model	Accuracy, %				Avg. time/epoch., sec
	epoch 30	epoch 50	ep 70	ep 100	
SQN	95.8	96.2	96.2	96.2	20.3
SGD	91.5	95.1	96.0	96.6	9.4

Conclusion

Our numerical results suggest that SQN converges faster than oLBFGS and SGD. By experimenting with neural networks, we demonstrated that SQN applicable for large-scale optimization problems.

References

[1] Richard H. Byrd, S. L. Hansen, Jorge Nocedal, and Yoram Singer. A stochastic quasi-newton method for large-scale optimization. 2016.

[2] Christopher Thiele, Mauricio Araya-Polo, and Detlef Hohl. Deep neural network learning with second-order optimizers - a practical study with a stochastic quasi-gauss-newton method. *CoRR*, 2020.

[3] Nicol N. Schraudolph, Jin Yu, and Simon Günter. A stochastic quasi-newton method for online convex optimization. 2007.

[4] K.D. Yakovlev.