Local Minima Shape for Artificial Neural Network

Adrien Bertaud, Yanis Bendali, Danya Li
École Polytechnique Fédérale de Lausanne, Switzerland

***Abstract*—The minima accessible by neural networks are studied. The concept of non-uniformity and sharpness are introduced, which characterize the shape of a minimum and hence the feasibility of convergence for a given optimizer. In particular, this work presents the influence of learning rate and batch size in local minima selection. Our emphasis will be SGD, GD, AdaGrad, and L-BFGS optimizers.**

I. Introduction

Interesting observation is that Stochastic Gradient Descent (SGD) tends to select flat-minima while Gradient Descent (GD) tends to select sharp-minima[2, 4, 6]. Many researches are conducted[1, 3] to show that not only optimizers, but also learning rate and batch size have influences on the selection of a minimum when training a network. Thus, theoretical analysis as well as experiments are performed from a dynamic stability perspective by Lei, Chao and Weinan[5] with focus on SGD and GD. Inspired by their work, we will try similar experiments in which we hope to verify their statements firstly and apply them to other algorithms such as Adaptive Gradient (AdaGrad) and Limited-memory BFGS (L-BFGS) optimizers secondly.

In this report, we start from introducing two essential quantities which determine the shape of a minima. Afterwards, we explain the configuration of our neural network experiments. Then we study the influence of learning rate as well as batch size on algorithms of interest.

II. Models and Methods

A. Definition of sharpness and non-uniformity

Let's first consider a training loss to minimize defined as:

$$f(x) = \frac{1}{n}\Sigma_{i=1}^{n} f_i(x)$$

When training a network with this loss, the optimizer should help us converge to a minimum point x^* such that :

$$\nabla f(x^*) = \frac{1}{n}\Sigma_{i=1}^{n} \nabla f_i(x^*) = 0$$

At this point, we define the hessian H_i of each function f_i. Then, we can define Hessian matrix and variance matrix that will be useful for the characterisation of minima shape as:

$$H = \frac{1}{n}\Sigma_{i=1}^{n} H_i$$
$$\Sigma = \frac{1}{n}\Sigma_{i=1}^{n} H_i^2 - H^2$$

Thus, taking the maximum eigenvalue of each matrix, *sharpness* and *non-uniformity* can be defined respectively, as follows:

$$s = \lambda_{max}(H) \quad (1)$$
$$\nu = \lambda_{max}(\Sigma^{1/2}) \quad (2)$$

The computation of sharpness and non-uniformity for a given set of training parameters then needs a way to find the maximum eigenvalue of a given matrix. A popular way to do it is to use the power method algorithm. It is an iterative algorithm that takes a diagonalizable matrix H with a dominant eigenvalue λ_{max} such that $|\lambda_{max}|$ is greater than the absolute values of all the other eigenvalues of H and an initial nonzero vector x_0. Then, we can construct two sequences (x_k) and (λ_k) such that $x_{k+1} = \frac{Hx_k}{||Hx_k||_2}$ and $\lambda_k = x_k^T H x_k$. The sequence (λ_k) is then defined to converge to λ_{max}. In our case, this construction is relevant because we consider positive semidefinite matrices (H and Σ) whose eigenvalues are non-negative.

In fact, it is also possible to prove a theoretical result of convergence when the optimizer used is SGD or GD[5] :

$$0 \leq s \leq \frac{2}{\eta} \quad (3)$$
$$0 \leq \nu \leq \frac{1}{\eta}\sqrt{\frac{B(n-1)}{n-B}} \quad (4)$$

This condition of convergence is very interesting because it allows us to build upper bounds for the two important quantities that the optimizer will be able to converge for a given learning rate and batch size (when they are fixed).

B. Neural network experiments

An important aspect of neural network is its non-convexity which means a huge number of local minima.

For further exploration, Fashion MNIST dataset with 60,000 training data and 10,000 test data is used. However, as computation for convergence and non-uniformity is too expensive, we restricted example size of 1,000 for training.

Thus, for SGD, batch size of 1,000 is exactly GD in our case. The network structure we use is presented in Table I.

Layer	Parameters
linear + ReLU	784 $\rightarrow$ 256
linear + ReLU	256 $\rightarrow$ 256
linear + ReLU	256 $\rightarrow$ 256
linear + ReLU	256 $\rightarrow$ 256
linear	256 $\rightarrow$ 10

Table I: Neural network for Fashion MNIST

To train the neural network, we considered different optimizers to minimize the cross entropy loss. We used SGD, GD, AdaGrad and L-BFGS, and we tried to characterized the shape of the minima as they converge with fixed learning rate or batch size.

Training process stops only if the maximum number of iteration of 20,000 is reached or a training loss is less than 10^{-4}. To compute the experimental sharpness and non-uniformity, we applied the power method algorithm to the estimation of the loss function with the training data set.

III. Results

A. Results for SGD

We began to verify what is done in the paper[5] for SGD and GD. So we trained models with different combinations of learning rate and batch size using SGD optimizer. Results are presented in Fig.1.

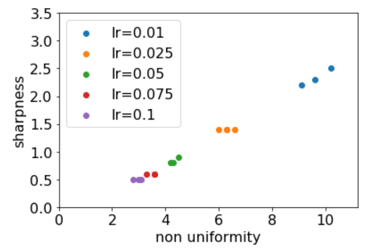
As expected in Fig.1a, no matter what the learning rate is, sharpness and non-uniformity of selected minima never reach the theoretical limits calculated in Eq.3-4. We can notice a linear relationship between sharpness and non-uniformity in both Fig.1a and Fig.1b. The quantities are well clustered in the order of learning rate or batch size.

B. Results for AdaGrad

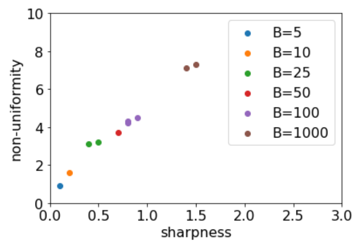
We are now considering the case of AdaGrad as a training optimizer (Fig.2).

The first thing to note is the similarity of the non-uniformity/sharpness curve we obtain compared to the one for SGD in Fig.2a and Fig.2b. We noticed that the relationship between these two quantities also seems to be a proportional relationship just like for SGD. The noticeable difference here is that the different points we consider are not clustered by batch size. Hence, it seems that the batch size has less influence on sharpness and non-uniformity for AdaGrad.

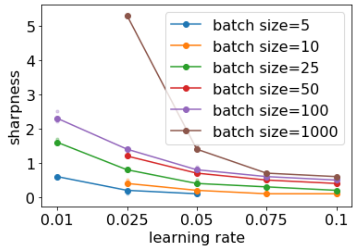
Moreover, comparing the influence of batch size and learning rate for the sharpness, we notice that sharpness tends



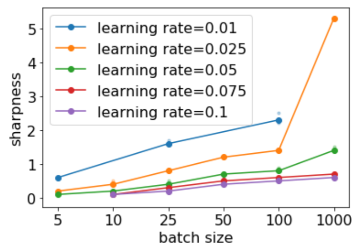
(a) Sharpness vs non-uniformity with different learning rates



(b) Non-uniformity vs sharpness with different batch sizes

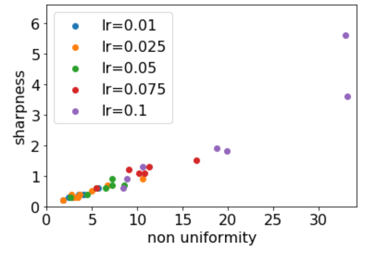


(c) Sharpness vs learning rate with different batch size

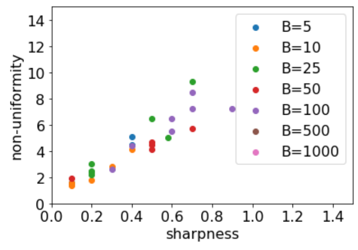


(d) Sharpness vs batch size with different learning rates

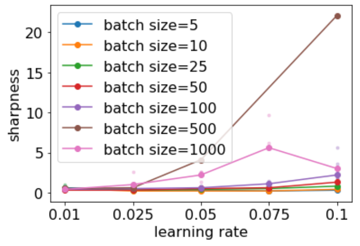
Figure 1: Results for SGD. Training results for each combination of parameters are displayed with very small light points while the mean values are displayed with big circles and lines.



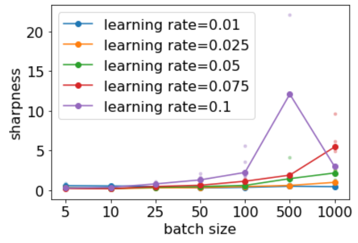
(a) Sharpness vs non-uniformity with different learning rates



(b) with different batchsizes



(c) Sharpness vs learning rate



(d) Sharpness vs batch size

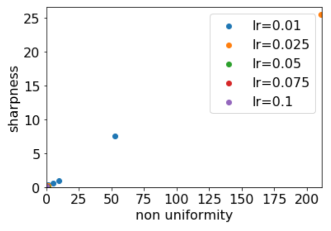
Figure 2: Results for AdaGrad.

to increase as the batch size increases (Fig.2d) like SGD. However unlike SGD, sharpness would also increase with the learning rate (Fig.2c). This strange behaviour is probably linked to the adaptive behaviour of AdaGrad where it decreases the learning rate while converging.

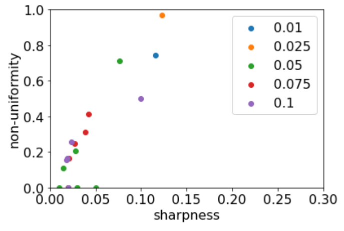
C. *Results for L-BFGS*

Finally, we consider the minima selection for the L-BFGS optimizer (Fig.3). Because of its definition, it has been used when the batch size is equal to the total number of training samples just as GD.

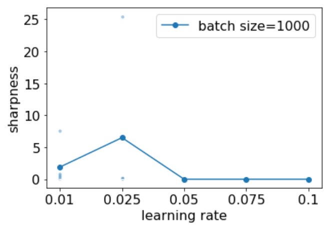
With Fig.3a and Fig.3b, it is possible to notice that L-BFGS tends to satisfy a linear relationship between sharpness and non-uniformity as well as for SGD and AdaGrad. It seems that L-BFGS also allows to reach higher values of non-uniformity and sharpness in our simulations (Fig.3d). However, as it is shown in Fig.3c, learning rate does not seem to have a real influence on the sharpness of the convergence point of training. It would be interesting to make additional tests to have a more explicit curve shape.



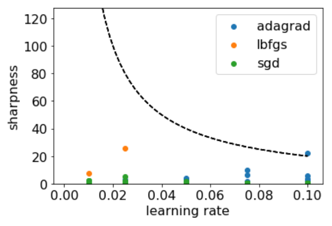
(a) L-BFGS sharpness vs non-uniformity with different learning rates



(b) L-BFGS with different learning rates



(c) L-BFGS sharpness vs learning rate



(d) Sharpness vs learning rate with all batch sizes, dash line representing sharpness criterion limit for SGD

Figure 3: Results for L-BFGS.

IV. Discussion

For SGD and GD, the tendencies and the evolution of sharpness and non-uniformity depending on the learning rate and batch size are as expected.

For a given algorithm, the non-uniformity is always linear relatively to sharpness. For that the following remarks focus only on sharpness. And this could also allow to concentrate future studies on sharpness as it is faster to compute than non-uniformity.

In Fig.3d, we can see that SGD sharpness is always under the theoretical convergence limit, as expected. For AdaGrad and L-BFGS, the theoretical limit of sharpness and non-uniformity cannot be applied anymore.

AdaGrad shows a very low sharpness for low learning rates (inferior to 0.075), but has a higher sharpness for high learning rates (superior to 0.075). We expected to have no significant influence of learning rate because it is adapted. We are also surprised that the tendency is at the opposite of SGD, with higher learning rate giving higher sharpness. We can intuitively understand why AdaGrad has higher sharpness with an higher learning rate because the learning rate is adaptive and is reduced during training. So starting with a low learning rate will reduce it too fast.

For GD and SGD the sharpness is well clustered to learning rate or batch size. At the opposite, for AdaGrad and L-BFGS, the sharpness is not well clustered. This means that learning rate and batch size seems to be relevant only to predict shape with GD and SGD but not with AdaGrad and L-BFGS.

V. Summary

The aim of this work was to notice the main differences of minima selection between SGD and other optimizers. SGD shows strong theoretical limits that allows us to build upper bound on sharpness and non-uniformity for a given step size and batch size. AdaGrad and L-BFGS are optimizers that allows to have stronger convergence rates but their minima selection do not present simple correlations with the initial hyperparameters.

References

[1] Priya Goyal et al. “Accurate, large minibatch sgd: Training imagenet in 1 hour”. In: *arXiv preprint arXiv:1706.02677* (2017).

[2] Sepp Hochreiter and Jürgen Schmidhuber. “Flat minima”. In: *Neural Computation* 9.1 (1997), pp. 1–42.

[3] Elad Hoffer, Itay Hubara, and Daniel Soudry. “Train longer, generalize better: closing the generalization gap in large batch training of neural networks”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 1731–1741.

[4] Nitish Shirish Keskar et al. “On large-batch training for deep learning: Generalization gap and sharp minima”. In: *arXiv preprint arXiv:1609.04836* (2016).

[5] Lei Wu, Chao Ma, and E Weinan. “How sgd selects the global minima in over-parameterized learning: A dynamical stability perspective”. In: *Advances in Neural Information Processing Systems*. 2018, pp. 8279–8288.

[6] Lei Wu, Zhanxing Zhu, et al. “Towards understanding generalization of deep learning: Perspective of loss landscapes”. In: *arXiv preprint arXiv:1706.10239* (2017).