Mini-batch Size and Convergence Diagnostics for SGD

Mini-project for CS-439 Optimization for Machine Learning

Atzeni Mattia[1], Cloux Olivier[2], and Janchevski Andrej[2]

[1]Department of Electrical Engineering, EPFL
[2]Department of Computer Science, EPFL

***Abstract*—In this paper we present our research on the topic of optimizers suitable to tackle deep learning tasks. In particular, we extend the knowledge about a recent modification of SGD called $\texttt{SGD}^{1/2}$. We will show the influence of mini-batch size on the training time and accuracy, with a comparison across other optimizers. Finally, we will discuss the different properties of the optimization algorithms with respect to their convergence behaviour. The results proved we can recommend higher mini-batch sizes for adaptive learning rate optimization and we support the utilization of the improved SGD for its convergence benefits.**

I. Introduction

With the rise of machine learning and artificial intelligence in the last decades, several domains in science, society and technology have been disrupted. However, more data and more powerful models always incur a cost: power. Training deep learning models has become exponentially harder, as we now have access to more data than ever and use more complex architectures. This leads either to longer training time, or to the necessity to use a more powerful and costly computing infrastructure.

In this report, we explore methods to accelerate the training of such models, using multiple techniques, such as adaptive learning rates and convergence diagnostic measures. In particular, we investigate the performance of a recent optimization algorithm, $\texttt{SGD}^{1/2}$, a variant of stochastic gradient descent (SGD) introduced by Chee and Toulis [1]. In this regard, our experiments aimed at extending previous work by:

- comparing the performance of $\texttt{SGD}^{1/2}$ to well-known optimization algorithms like SGD and Adam [2];
- estimating the convergence time of these optimizers using diagnostic measures like the Pflug diagnostic [1], [3] and modelling the respective convergence region;
- exploring the influence of the mini-batch size and the learning rate on the performance of these optimization algorithms through an extensive grid search on the two parameters.

We performed our experiments both on synthetic data and on well-known datasets such as MNIST [4] and Fashion-MNIST [5]. We employed a simple feed-forward neural network on the synthetic data and a convolutional neural network (CNN) on MNIST and Fashion-MNIST. We hypothesized and wished to confirm that similar results could be obtained for these more complex classification problems compared to results on the synthetic data.

II. Methods

A. Convergence estimation

As mentioned above, for our analysis we experimented using $\texttt{SGD}^{1/2}$, a variant of SGD recently introduced by Chee and Toulis in [1]. This algorithm relies on a diagnostic measure computed from iteration data during training, called Pflug's diagnostic [1], [3].

The Pflug diagnostic attempts to estimate the iteration at which an iterative stochastic algorithm has converged. The diagnostic is iteratively calculated as an approximation to the expected value of the gradient orientation. Specifically, the formulas for initializing the diagnostic and the iterative step for its update are presented in Equation (1):

$$s_0 = 0,\ s_n = s_{n-1} + \nabla\theta_n^T \nabla\theta_{n-1}. \tag{1}$$

This criterion is based on the assumption that the training process is split into two phases. During the initial 'transient' phase, the diagnostic value is positive, indicating that the parameter value iterates have not yet reached the optimum, as the gradients are still similarly orientated. While on the other hand, the moment the diagnostic value drops below 0, it is said that the training has reached the 'stationary' phase and the iterates are said to randomly oscillate in a small region around the optimum. After this point, successive gradients are severely misaligned.

The $\texttt{SGD}^{1/2}$ optimization algorithm wishes to take advantage of this diagnostic during the iterative parameter update step, in order to perform early stopping in the case convergence has already been detected. Specifically, this SGD variant dynamically reduces the learning rate by half each time iterates are said to have entered the 'stationary' phase, with a subsequent re-initialization of the diagnostic value. When the learning rate has been diminished significantly, the procedure can be completely halted. By definition, the procedure also automatically updates a separate control variable that refers to the iteration where convergence was detected last. The exact pseudo-code for this algorithm, as presented in the original paper, is provided in the Appendix, in Algorithm 1.

B. Datasets

Three different datasets were used for training the models using the three optimization algorithms. The first dataset was

automatically generated for the purpose of training and evaluation of a simple 2D quadratic-boundary classifier. The data is a set of points uniformly sampled from the $[-1, 1]^2$ space and a point has the positive class if it lies inside a circle with radius $\sqrt{\frac{2}{\pi}}$. Besides this synthetic data, we utilized two other datasets, namely the ten-class MNIST and Fashion-MNIST collections of images. To reduce computational complexity, these two datasets were down-sampled to contain only 1000 training and testing images each. We performed stratified sampling to respect the original class distribution.

C. Models

We employed two different model types, one for the synthetic 2D circle dataset and another one for MNIST and Fashion-MNIST. The classification model for the circle dataset was a simple neural network with one hidden layer containing 32 units, with ReLU as the activation function. In order to preserve the relevance of the comparison, both the MNIST and Fashion-MNIST models used the same deep architecture in their design: a standard CNN with 3 convolutional layers interleaved with batch normalization and 2 fully-connected layers. A more detailed graphical visualization of the architecture and image transformations performed by this model is available in Figure 3 in the Appendix.

III. Experiments

For this project we wished to execute mainly two different kinds of experiments. For the first experiment, our goal was to expand on the research by the authors of $\texttt{SGD}^{1/2}$ [1], by first investigating the influence of choosing different mini-batch sizes, learning rates and loss functions. We compared the results with already established optimization algorithms such as standard SGD and Adam.

For the second experiment, we wished to run simulations alike the ones in [1], in order to not only estimate the shape of the Pflug convergence region, but also determine its possible dependence on the loss function and the optimizer.

For each experimental condition (combination of choice of dataset, optimizer and loss function), each model was trained on a dataset with 900 samples, evaluated on a separate validation set with 100 samples and on an independent test set with 1000 samples. Mini-batch sizes were selected from a geometric progression of 42 values in the range $[1, 900]$. This sequence of values allows us to transfer smoothly from the standard single-sample SGD to using standard batch gradient descent (i.e., computing parameter updates based on the values of all training samples grouped together in a single batch; when all training samples are used in the batch in falls back to standard GD).

Similarly, the initial learning rates were selected from a geometric progression of 10 values in the range $[0.01, 1]$. From careful observations in practice, the initial range of learning rate values required adjustment for different optimization algorithms and loss functions. Adam required a down-scaling of the learning rate with a factor of 0.05, while the values were up-scaled by a factor of 5 for $\texttt{SGD}^{1/2}$. MSE loss had also shown to require an increase of the learning rate range by a factor of 5.

During training, the loss value, accuracy and F_1 metric were computed for the training and validation data respectively. In addition, for the first experiment, these metrics were also computed for the independent test data. From the extracted sequence of parameter gradient values computed after each training iteration, the Pflug diagnostic was calculated and used to estimate the epoch at which convergence was reached. For the second experiment, we had chosen to empirically estimate the convergence region as an ellipse, whose center and two axes were calculated in a similar fashion to the procedure employed in [1]. Specifically, 1000 simulations of the model training were run for each experiment condition, to vary the initialization of the model parameters. The empirical estimates were based on a 95% confidence interval on values from the first 2 parameters of each model (in order for 2D visualization of the region to be possible).

IV. Results

In Figure 1 we present the results from our first experiment related to the mini-batch size. In the first two rows we compare (1) the achieved prediction performance[1] and (2) the total training time in seconds for different mini-batch sizes and across the three optimization algorithms (SGD, Adam and $\texttt{SGD}^{1/2}$). In each plot, we carry out the comparison across both datasets and loss functions. We can observe that: the simplest 2D circle problem is easiest to solve, while the image datasets as expected proved to be more challenging; SGD seems to have a specific range of optimal mini-batches sizes, while Adam and $\texttt{SGD}^{1/2}$ require the largest mini-batches, as this would provide them with more accurate gradient information (less stochasticity). The figures for the $\texttt{SGD}^{1/2}$ contain more noise, as this optimizer also performs early-stopping, leading to more unpredictable results. Interestingly, since we leveraged GPUs for training, higher mini-batch sizes result in exponentially lower training times. This motivated us to consider both of these quantities in order to gauge the optimal mini-batch sizes. A combined score was calculated as a simple difference between the test F_1 and training time, both of these quantities normalized with standard scaling across experiment conditions. The graphs for this combined score are presented in the last row and we observe that this criterion is less dependent on the choice of the dataset. As such, this score was used to discover the optimal mini-batch sizes and learning rates for each experiment condition and these optimal values are displayed in Table I in the Appendix.

Regarding the learning rates, our experiment confirmed that the Pflug diagnostic is extremely sensitive to the exact value for this hyperparameter, as also noted in [1]. In Figure 4 we show the relationship between the re-scaled learning rates and the convergence threshold. As expected, it can be observed that for most experiment conditions very small values lead to a later convergence and also the converse.

[1] F_1 score on the independent test set

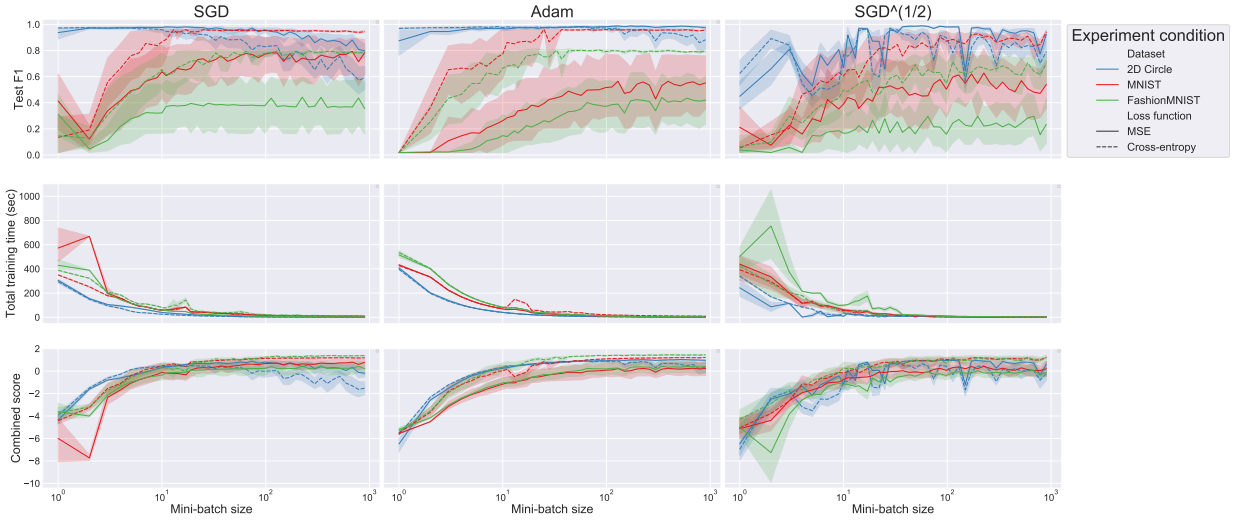

Fig. 1. Mini-batch size versus F_1 score on the test set (top), training time (middle) and combined F_1/time score (bottom) across optimizers

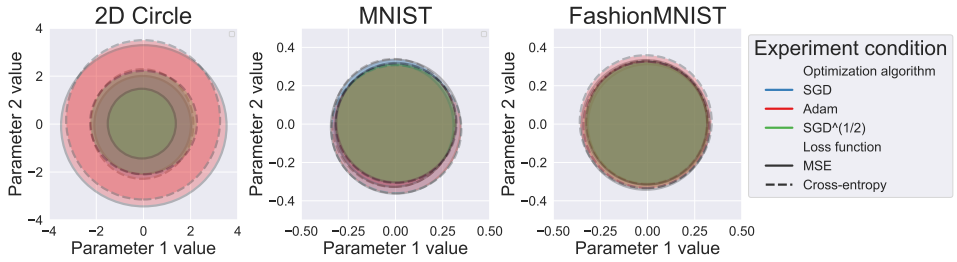

Fig. 2. Visualization of the estimated convergence regions across datasets

As the authors also stated, with Figure 5 we show very efficiently the reason why a validation set is sometimes ineffective in estimating the convergence threshold. As we can notice, when the evaluation data originates from a similar distribution as the training data, powerful classifiers such as the neural networks we utilized have very good generalization. Validation accuracy follows the same trend as its respective training value.

In Figure 2 we present the results from our second experiment related to the shape of the convergence regions. Along the row the ellipses are compared across the different datasets and each plot compares across optimizers and loss functions. From these results we present the main benefit of the $\texttt{SGD}^{1/2}$ algorithm - its main trademark is the reduction of the area of the convergence region to a smaller size compared to the others. The Adam algorithm on the other hand seems to increase the radius around the optimum. These effects, as we observe, are more pronounced for the simple 2D classification dataset, while MNIST and Fashion-MNIST seem to have a tighter bound on their respective minima.

V. Conclusion

For this project, we explored methods to accelerate the training of neural network models, using multiple techniques such as early stopping and adaptive learning rates. In particular, we investigated the performance of a recent optimization algorithm, $\texttt{SGD}^{1/2}$, a variant of stochastic gradient descent (SGD). In this regard, our experiments compared the performance of $\texttt{SGD}^{1/2}$ to well-known optimization algorithms like SGD and Adam, estimated the convergence of these optimizers using diagnostic measures like the Pflug diagnostic and explored the influence of the mini-batch size and the learning rate on the performance of these optimization algorithms.

The results suggested that when more powerful hardware like GPUs is available to researchers, larger mini-batches should be utilized for adaptive learning rate algorithms such as Adam and $\texttt{SGD}^{1/2}$, in order to leverage both prediction accuracy and computational cost, while for the standard SGD the mini-batch size should still be tuned. We were able to confirm that the hypotheses presented in the authors' paper hold as strongly for deep architectures and that the main benefit of the improved SGD is the reduction in the convergence region, while a small downside might prove to be the unpredictability of the prediction performance.

Acknowledgements

We wish to thank the LTS2 laboratory for lending us their GPU rig, enabling us to train non-trivial models and run our experiments in a reasonable time.

REFERENCES

[1] J. Chee and P. Toulis, "Convergence diagnostics for stochastic gradient descent with constant learning rate," in *International Conference on Artificial Intelligence and Statistics, AISTATS 2018*, ser. Proceedings of Machine Learning Research, vol. 84. PMLR, 2018, pp. 1476–1485.

[2] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[3] G. C. Pflug, "Gradient estimates for the performance of markov chains and discrete event processes," *Ann Oper Res*, vol. 39, no. 1, pp. 173–194, Dec. 1992. [Online]. Available: https://doi.org/10.1007/BF02060941

[4] Y. LeCun, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/.

[5] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017.

[6] "NN SVG." [Online]. Available: http://alexlenail.me/NN-SVG/index.html

APPENDIX A
ALGORITHMS

Algorithm 1 Procedure $\texttt{SGD}^{1/2}$

Input: θ_0, data $\{(y_1, x_1), (y_2, x_2), \ldots\}$, $\gamma > 0$, $\texttt{burnin}$, $\texttt{maxit} > 0$

Output: Iteration $\tau > 0$, when SGD is estimated to have converged.

$s \leftarrow 0$
$\tau \leftarrow 0$
$\theta_1 \leftarrow \theta_0 - \gamma \nabla \ell(y_1, x_1^\top \theta_0)$
for all $n \in \{2, 3, \ldots\}$ **do**
  $\theta_n \leftarrow \theta_{n-1} - \gamma \nabla \ell(y_n, x_n^\top \theta_{n-1})$
  $s \leftarrow s + (\theta_n - \theta_{n-1})^\top (\theta_{n-1} - \theta_{n-2})/\gamma^2$
  if $n > \tau +$ $\texttt{burnin}$ and $s < 0$ **then**
    $\tau \leftarrow n$
    $s \leftarrow 0$
    $\gamma \leftarrow \gamma/2$
    if $\gamma < 10^{-10}$ and $n >$ $\texttt{maxit}$ **then**
      return θ_n.
    end if
  end if
end for

APPENDIX B
TABLES

TABLE I
OPTIMAL MINI-BATCH SIZE AND LEARNING RATES DISCOVERED BY GRID SEARCH FOR EACH DATASET, OPTIMIZER, AND LOSS FUNCTION

Dataset	Optimizer	Loss function	Optimal mini-batch size	Optimal learning rate
2D Circle	Adam	Cross Entropy	682	0.050
		MSE	517	0.150
	SGD	Cross Entropy	113	1.000
		MSE	113	1.077
	$\texttt{SGD}^{1/2}$	Cross Entropy	225	1.797
		MSE	113	8.985
MNIST	Adam	Cross Entropy	900	0.011
		MSE	784	0.004
	SGD	Cross Entropy	900	0.599
		MSE	297	1.797
	$\texttt{SGD}^{1/2}$	Cross Entropy	85	0.083
		MSE	129	1.936
Fashion-MNIST	Adam	Cross Entropy	900	0.011
		MSE	341	0.007
	SGD	Cross Entropy	900	0.215
		MSE	682	0.387
	$\texttt{SGD}^{1/2}$	Cross Entropy	225	0.646
		MSE	85	0.417

APPENDIX C
FIGURES

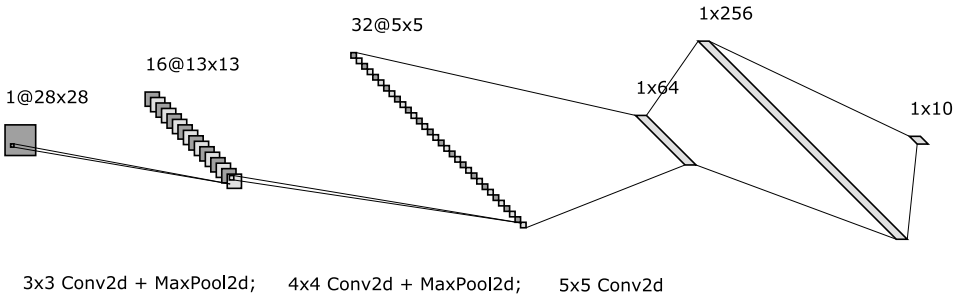

Fig. 3. Visualization of the ConvNet structure of the deep model (using tool from [6]). The architecture design followed a standard CNN pattern: 16 convolution channels of kernel-size 3, 32 convolution channels of kernel-size 4, 64 convolution channels of kernel-size 5, with finally classifying the output using two fully-connected layers of 256 and 10 units respectively. Intermediate layers performing max-pooling with a kernel size and stride of 2, ReLU activation and batch normalization are also present.

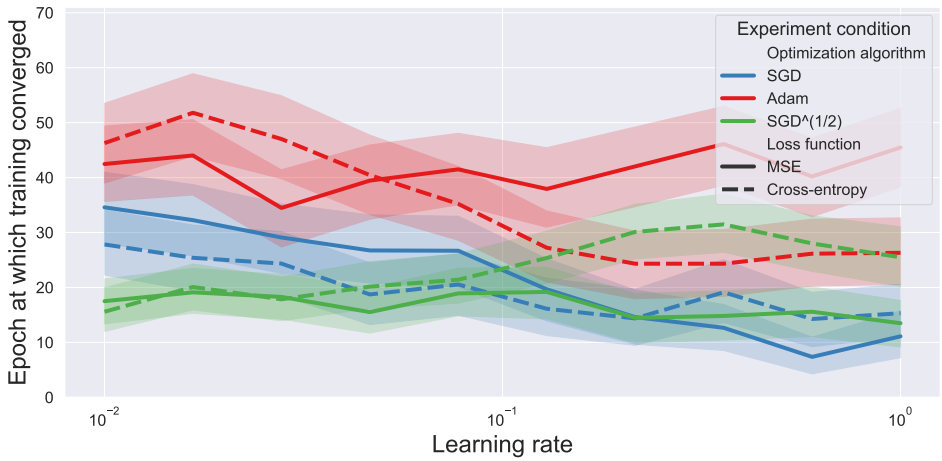

Fig. 4. Learning Rate Versus Convergence Time

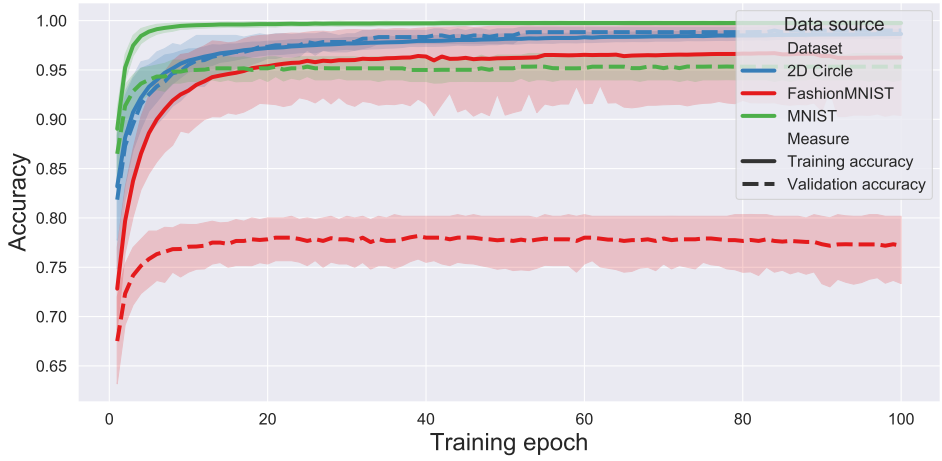

Fig. 5. Training Accuracy Versus Validation Accuracy