American Journal of Computer Science and Technology
2021; 4(4): 90-96
http://www.sciencepublishinggroup.com/j/ajcst
doi: 10.11648/j.ajcst.20210404.11
ISSN: 2640-0111 (Print); ISSN: 2640-012X (Online)

Optimizing the Hyper-parameters of Multi-layer Perceptron with Greedy Search

Mingyu Bae

North London Collegiate School Jeju, Jeju, Korea

Email address:
mgbae22@pupils.nlcsjeju.kr

To cite this article:
Mingyu Bae. Optimizing the Hyper-parameters of Multi-layer Perceptron with Greedy Search. *American Journal of Computer Science and Technology*. Vol. 4, No. 4, 2021, pp. 90-96. doi: 10.11648/j.ajcst.20210404.11

Received: September 13, 2021; **Accepted**: October 4, 2021; **Published**: October 15, 2021

Abstract: The core of deep learning network is hyper-parameters which are updated through learning process with samples. Whenever a sample is fed into deep learning network, parameters change according to gradient value. At this point, the number of samples and the amount of learning are crucial, which are batch size and learning rate. To find the optimal batch size and learning rate, lots of trial is inevitable so it takes so much time and effort. Therefore, there have been lots of papers to enhance the efficiency of its optimization process by automatically tuning the single parameter. However, global optimization can't be guaranteed by simply combining separately optimized parameters. This paper propose brand new effective method for hyperparameter optimization in which greedy search is adopted to find the optimal batch size and learning rate. In experiment with Fashion MNIST and Kuzushiji MNIST dataset, the proposed algorithm shows the similar performance as compared to complete search, which means the proposed algorithm can be a potential alternative to complete search.

Keywords: Hyper Parameters, Batch Size, Learning Rate, Greedy Search

1. Introduction

The rise of deep learning has the potential to transform our future even more than it already has and perhaps more than any other technology. Deep learning has already created significant improvements in computer vision, speech recognition, and natural language processing, which has led to deep learning based commercial products being ubiquitous in our society and in our lives.

The recent decade saw dramatic success of deep neural networks based on the optimization method of stochastic gradient descent (SGD) [1, 2]. It is an interesting and important problem that how to tune the hyper-parameters of SGD to make neural networks generalize well. Some works have been addressing the strategies of tuning hyper-parameters [3-6] and the generalization ability of SGD [7-11]. However, there still lacks solid evidence for the training strategies regarding the hyper-parameters for neural networks.

Mini-batch stochastic gradient descent and variants thereof have become standard for large-scale empirical risk minimization like the training of neural networks. These methods are usually used with a constant batch size chosen by simple empirical inspection. The batch size significantly influences the behavior of the stochastic optimization algorithm, though, since it determines the variance of the gradient estimates. This variance also changes over the optimization process; when using a constant batch size, stability and convergence is thus often enforced by means of a (manually tuned) decreasing learning rate schedule.

Learning rate decay is a de facto technique for training modern neural networks. It starts with a large learning rate and then decays it multiple times. It is empirically observed to help both optimization and generalization. Common beliefs in how lrDecay works come from the optimization analysis of SGD: 1) an initially large learning rate accelerates training or helps the network escape spurious local minima; 2) decaying the learning rate helps the network converge to a local minimum and avoid oscillation. Despite the popularity of these common beliefs, experiments suggest that they are insufficient in explaining the general effectiveness of lrDecay in training modern neural networks that are deep, wide, and nonconvex.

Currently there are no simple and easy ways to set hyper-parameters – specifically, learning rate, batch size, momentum, and weight decay. A grid search or random search [12] of the hyper-parameter space is computationally expensive and time consuming. Yet training time and final performance is highly dependent on good choices.

In this paper, a new approach based on greedy search is proposed which is less time consuming but shows affordable performance.

2. Related Works

2.1. Batch Size

Some papers investigate how batch sizes affect the performance of deep learning networks. Batch size is known as the most important parameters in deep learning networks because it affects gradient direction. Now that GPU can be used in training, the batch size is also influenced by GPU memory and its architecture. However, there is no general rule of how to choose the proper batch size. Instead, it's determined through time-consuming repeated tests. Some researchers tried to automatically determine batch sizes by accumulating gradients of mini-batches and performing an optimization step at just the time when the direction of gradients starts to fluctuate [13].

2.2. Learning Rate

Learning rate decay is a de facto technique for training modern neural networks. It starts with a large learning rate and then decays it multiple times as shown in Figure 1. It is empirically observed to help both optimization and generalization. Common beliefs in how learning rate decay works come from the optimization analysis of Stochastic Gradient Descent. An initially large learning rate accelerates training or helps the network escape spurious local minima. It also helps the network converge to a local minimum and avoid fluctuation. Nevertheless, experiments suggest they are insufficient in explaining its general effectiveness in training neural networks. Regarding learning rate decay, there are several variants, but the fundamental idea is the same. Some research shows one can usually obtain the same learning curve on both training and test sets by instead increasing the batch size during training [14].

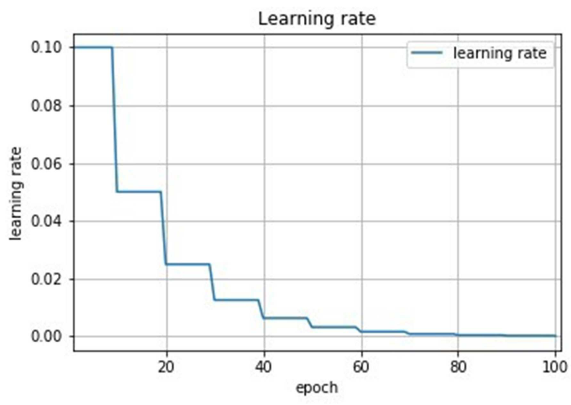


Figure 1. *Learning rate decay.*

Batch size and learning rate can be optimized separately but separate optimization doesn't guarantee the best performance when combined. Some papers have tried to tune these hyper-parameters at the same time [15, 16]. However, they're also based on empirical beliefs. The best practice can be the grid search or random search, but it requires lots of computation.

3. Proposed Method

The proposed method comes in the form of applying greedy search to find the best combination of batch size and learning rate.

3.1. Greedy Search

A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage. In many problems, a greedy strategy does not produce an optimal solution, but a greedy heuristic can yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.

For example, a greedy strategy for the "travelling salesman problem" is the following heuristic: "At each step of the journey, visit the nearest unvisited city." This heuristic does not intend to find the best solution, but it terminates in a reasonable number of steps; finding an optimal solution to such a complex problem typically requires unreasonably many steps. In mathematical optimization, greedy algorithms optimally solve combinatorial problems having the properties of matroids and give constant-factor approximations to optimization problems with the submodular structure.

3.2. Method

Because batch size and learning rate affect the performance of neural networks, Cartesian product of both parameters is used to find the best combination. For example, batch sizes={16, 32, 64, 128, 256} x learning rate={0.1, 0,05, 0,01, 0,005, 0,001} result in 25 combinations.

In this approach, either batch size or learning rate is fixed and then apply several instances of the other parameter. For instance, fixed learning rate, 0.1 with batch sizes={16, 32, 64, 128, 256} brings 5 cases among which the best case is chosen by measuring the accuracy with reserved test samples and then continue the same procedure until the end of epoch.

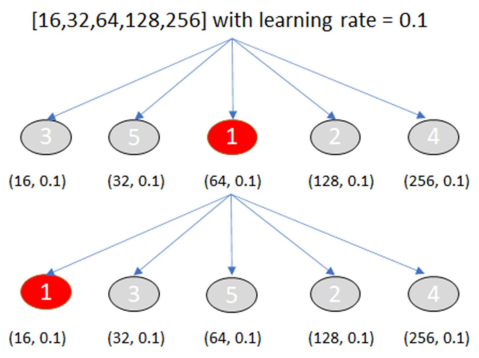


Figure 2. *Greedy Search.*

4. Experiment

In evaluating the performance of complete search and proposed method, the experimental settings are established like the following.

Training and test dataset: Fashion MNIST and Kuzushiji MNIST are used, both of which consist of 60,000 training samples and 10,000 test samples of 28 by 28 gray-scale pixel maps with 10 classes.

Neural network architecture: Multilayer perceptron (MLP) with Python. Its layer consists of 784, 256, 128, 100, 10 with sigmoid as the activation function.

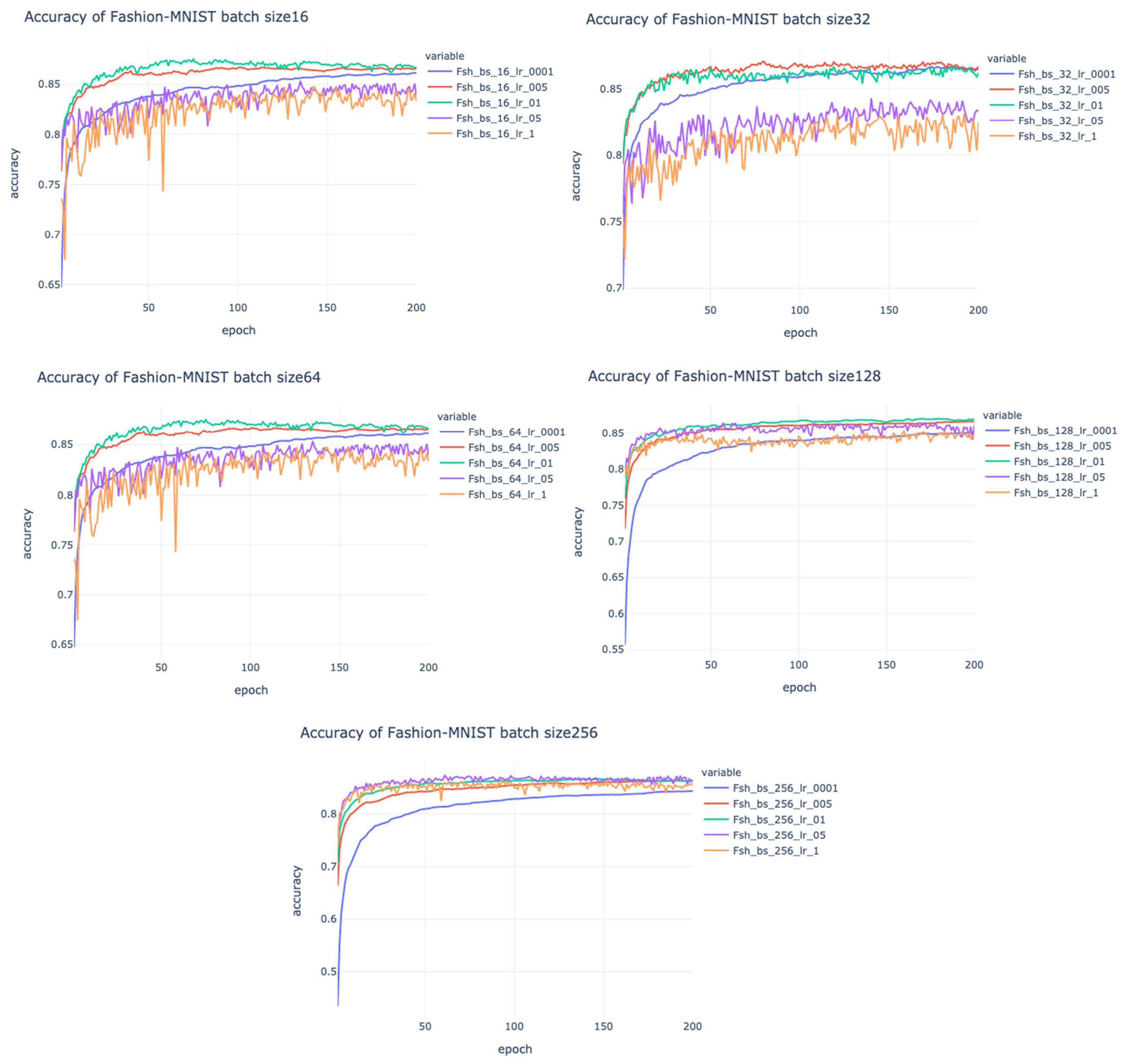
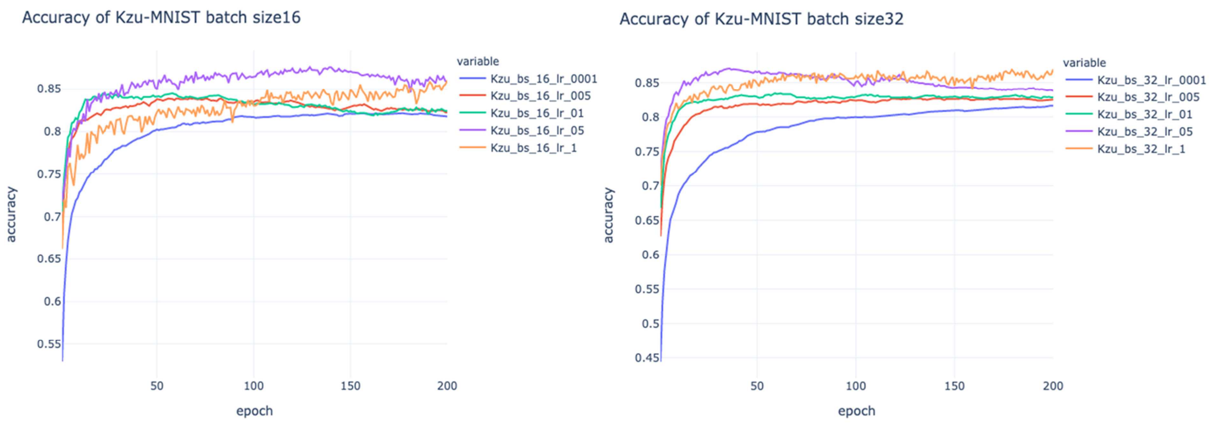


Figure 3. *Complete Search of Fashion MNIST.*



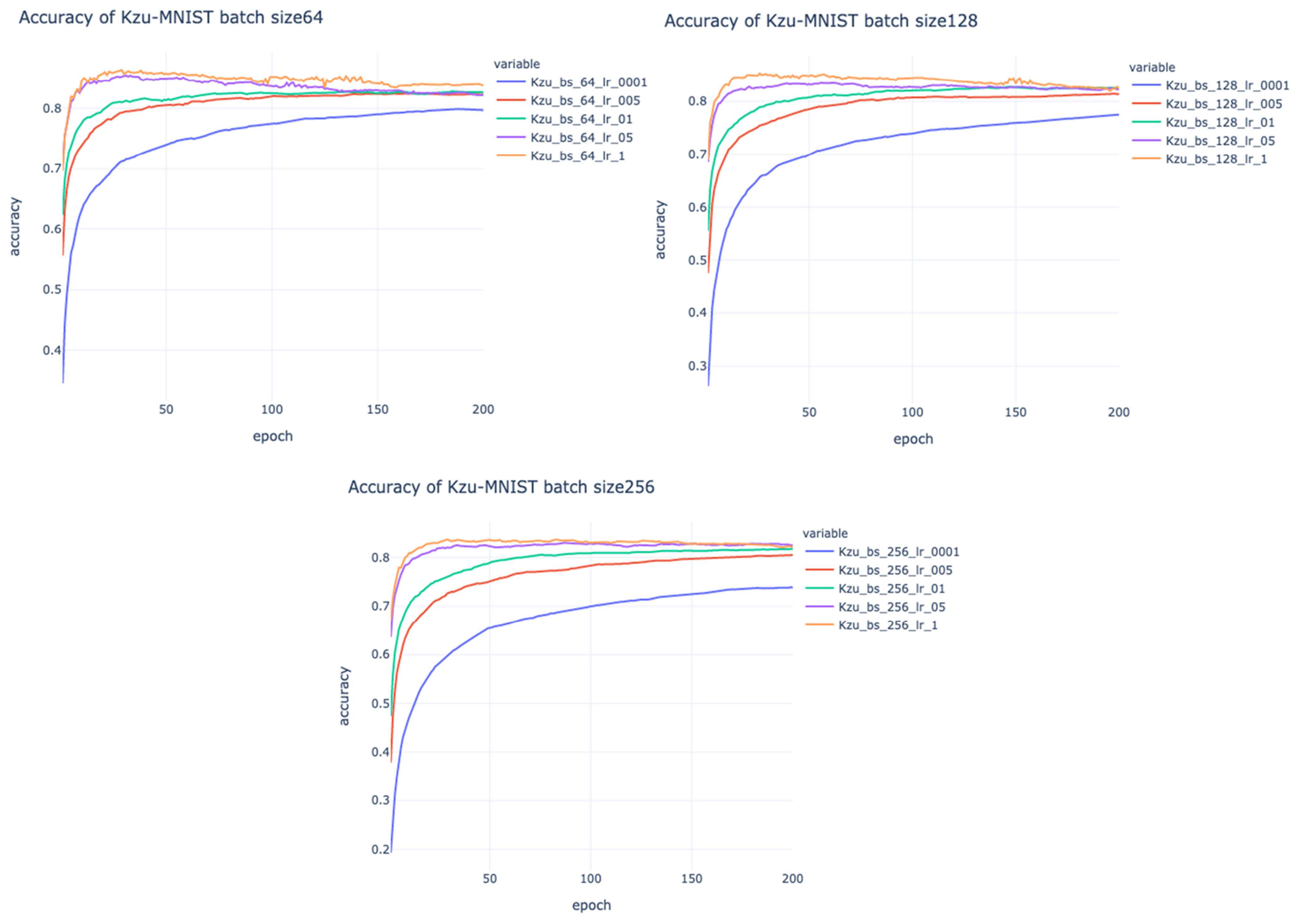


Figure 4. *Complete Search of Kuzushiji MNIST.*

4.1. Result of Complete Search

The results showed that using an extremely small or large learning rate decreased the accuracy of the complete search. For every batch size, either the learning rate of 0.001 or 0.1 had the lowest accuracy throughout the iteration. The result also showed the same tendency for the batch size. This proves that for MNIST, it's not recommended to increase or decrease the batch size and learning rate but rather keep it moderate.

The fluctuation of the accuracy was positively proportional to the learning rate and inversely proportional to the batch size. The blue curves that show the learning rate of 0.001 was almost a smooth curve whereas the orange curves that how the learning rate of 1 fluctuated a lot.

4.2. Result of Proposed Method

The complete search shows the accuracy varies depending on batch size and learning rate. Therefore, the best combination can be obtained only through lots of comparison experiments which is time-consuming and it's not feasible if dataset is huge.

In this paper, brand new optimization algorithm based on greedy search is proposed.

Training and evaluation dataset: Training samples is splitted into two sets, 90% training samples and 10% evaluation samples which are used to find the local best case.

Fixed batch size: When fixing batch size, there will be five cases according to learning rates, [0.1, 0.05, 0.01, 0.005, 0,001]. Greedy search chooses the best result among them every literation by measuring the accuracy with evaluation samples. Figure 5 and Figure 7 show five graphs for each fixed batch size. It's shown that the similar or the better accuracy rate as compared to complete search.

Fixed learning rate: When fixing learning rate, there will be five cases according to batch sizes, [16, 32, 64, 128, 256]. Likewise, Greedy search chooses the best result among them every literation by measuring the accuracy with evaluation samples. Figure 6 and Figure 8 show five graphs for each fixed learning rate. larger learning rate is expected to show fluctuation, but it depends on the characteristics of dataset. Kuzushiji MNIST doesn't show fluctuation while Fashion MNIST shows lots of fluctuation with higher learning rate, 0.1.

The result says it's not feasible to get the best combination through deterministic calculation. Therefore, efficient search like greedy is applicable in finding hyper-parameters.

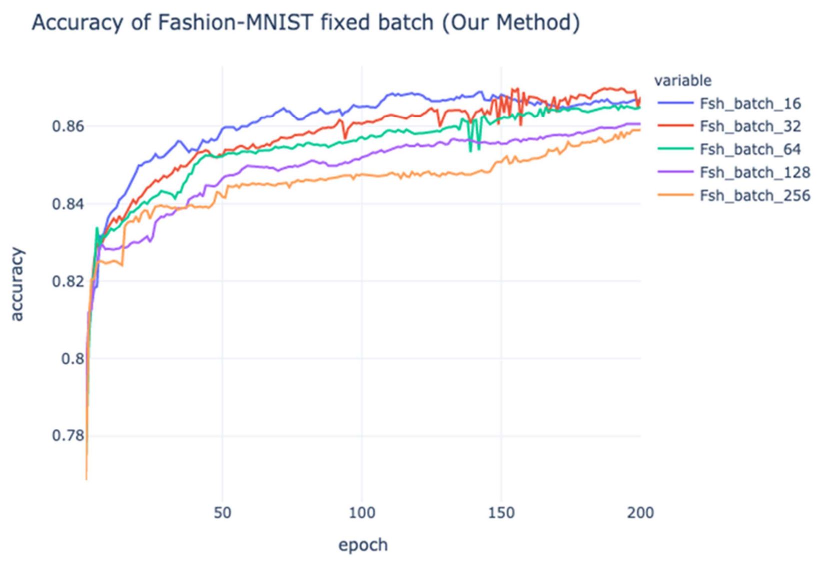


Figure 5. *Proposed method with fixed batch of Fashion MNIST.*

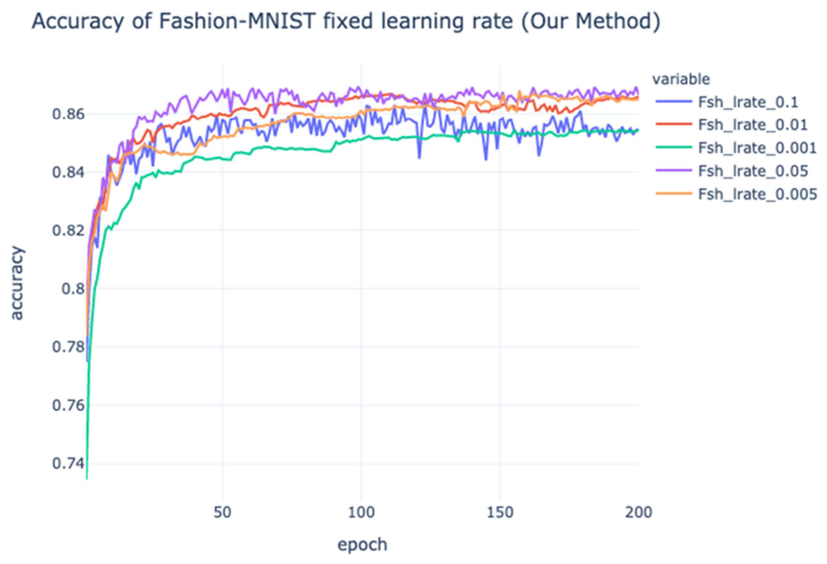


Figure 6. *Proposed method with fixed learning rate of Fashion MNIST.*

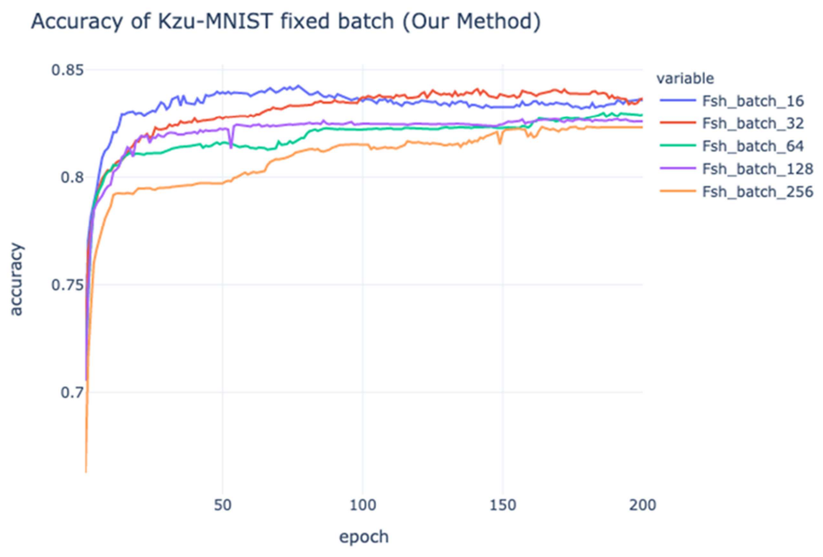


Figure 7. *Method with fixed batch of Kuzushiji MNIST.*

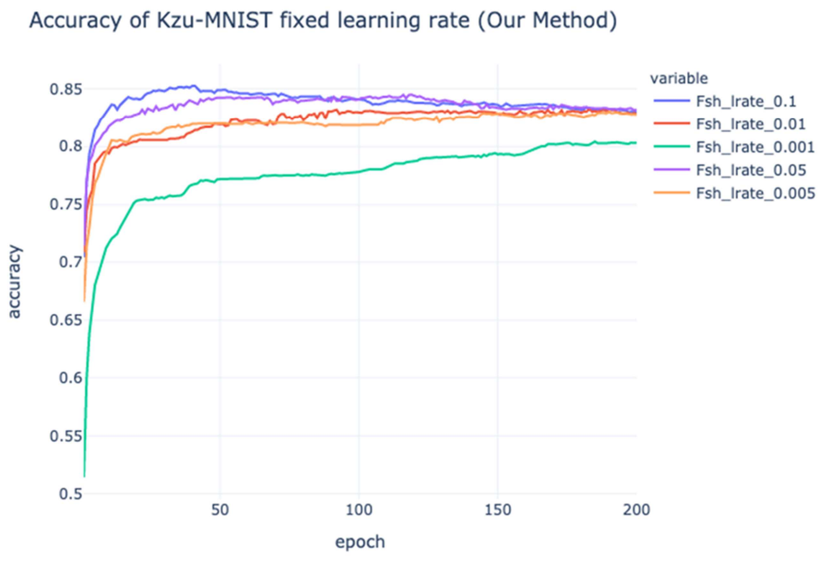


Figure 8. *Proposed method with fixed learning rate of Kuzushiji MNIST.*

5. Conclusion

In deep learning network, hyper-parameters are crucial because it directly affects the performance among which batch size and learning rate are the most important. This paper proposes brand new search algorithm to find the optimal combination of batch size and learning rate. Through comparison between complete search and proposed method, the proposed method shows its effectiveness. However, the proposed method doesn't guarantee global optimization either. Dependent upon dataset, the proposed method shows the better performance, but it doesn't in another dataset. And fixing learning rate shows the better performance as compared to fixing batch size because large learning rate may result in the fluctuation of accuracy.

However, if more parameters are chosen like batch size, 8, 16, 24, 32, up to 1024, complete search is not feasible because it'll take too much time. In such a case, the proposed method can be practically the alternative. As a further study, the method which considers both batch size and learning rate at the same time without fixing either of them during learning is needed to be investigated.

References

[1] L. Bottou. Online learning and stochastic approximations. On-line learning in neural networks, 17 (9): 142, 1998.

[2] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In International conference on machine learning, 2013.

[3] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio. Sharp minima can generalize for deep nets. In International Conference on Machine Learning, 2017.

[4] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: training imagenet in 1 hour. arXiv preprint arXiv: 1706.02677, 2017.

[5] S. Jastrzebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey. Three factors influencing minima in sgd. arXiv e-prints, 1711.04623, 2017.

[6] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In International Conference on Learning Representations, 2017.

[7] Y. Chen, C. Jin, and B. Yu. Stability and convergence trade-off of iterative optimization algorithms. arXiv preprint arXiv: 1804.01619, 2018.

[8] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In International Conference on Machine Learning, 2015.

[9] J. Lin, R. Camoriano, and L. Rosasco. Generalization properties and implicit regularization for multiple passes sgm. International Conference on Machine Learning, 2016.

[10] W. Mou, L. Wang, X. Zhai, and K. Zheng. Generalization bounds of sgld for non-convex learning: Two theoretical viewpoints. In Annual Conference On Learning Theory, 2018.

[11] A. Pensia, V. Jog, and P.-L. Loh. Generalization error bounds for noisy, iterative algorithms. In IEEE International Symposium on Information Theory, 2018.

[12] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, Feb, 2012.

[13] H Xu, J van Genabith, D Xiong, Q Liu, Dynamically Adjusting Transformer Batch Size by Monitoring Gradient Direction Change, arXiv preprint arXiv: 2005.02008, 2020.

[14] SL Smith, PJ Kindermans, C Ying, QV Le, DON'T DECAY THE LEARNING RATE, INCREASE THE BATCH SIZE, arXiv preprint arXiv: 1711.00489, 2017.

[15] L Balles, J Romero, P Hennig, Coupling adaptive batch sizes with learning rates, arXiv preprint arXiv: 1612.05086, 2016.

[16] F He, T Liu, D Tao, Control Batch Size and Learning Rate to Generalize Well: Theoretical and Empirical Evidence, Advances in Neural Information Processing Systems 32 (NeurIPS 2019).