ICLR 2019 REPRODUCIBILITY CHALLENGE AUTOLOSS: LEARNING DISCRETE SCHEDULE FOR ALTERNATE OPTIMIZATION

Parth Kothari, Yuejiang Liu, Timur Lavrov
Velvet Thunder Team, Machine Learning Course
École Polytechnique Fédérale de Lausanne
{parth.kothari,yuejiang.liu,timur.lavrov}@epfl.ch

ABSTRACT

AutoLoss: Learning Discrete Schedules for Alternate Optimization is a recent paper submitted to ICLR 2019, in which Xu et al. (2018) propose a meta-learning method to learn a schedule for alternate optimization and validates it through several empirical experiments. In this work, we aim to replicate the reported experiments in three tasks, namely d-ary regression, MLP classification and GANs. We first show that the results of standard baselines reported in their paper are reproducible. In addition, we demonstrate that hand-crafted schedules they present achieve competitive results. Subsequently, we provide a scenario where the proposed AutoLoss scheduler offers a performance boost. Finally, we discuss various challenges that we observed in their methodology that can lead to failed experiments.

1 INTRODUCTION

A fundamental and necessary part of research is to ensure that published results are reliable and reproducible. This, however, becomes particularly challenging in the machine learning community, as many factors like hyperparameters, random seeds, problem settings can have significant impact on algorithmic performance. In support of the ICLR 2019 Reproducibility Challenge, this work aims to replicate the experiments reported in a recent submission titled *AutoLoss: Learning Discrete Schedule For Alternate Optimization* Xu et al. (2018).

The key idea of AutoLoss is to determine the optimization schedule of various tasks through meta-learning. It learns and improves a discrete alternate schedule from the feedback of training and validation errors. Since objective functions are commonly composed of more than one term, for instance, a combination of task loss and regularization loss or a form of minimax game, the proposed method is compatible with many such tasks.

Our project aims to replicate the experiments from three different tasks reported in the paper under review. These consist of regression, classification and image generation. We report baseline results in two parts. First, we outline the results of performing standard grid search for the regularizer parameter which are highly in line with those reported in their paper. Second, we demonstrate that a well-tuned hand-crafted schedule can achieve a competitive performance in comparison to their proposal.

With respect to the proposed meta scheduler, we present a scenario where the proposed AutoLoss framework can provide a near-optimal schedule. As some of the hyperparameter settings and problem formulation are not disclosed yet, even after email exchanges with the authors, we found it highly challenging to reproduce all the results of their proposed methods. We thus close our experiments with a discussion on potential limitations of the AutoLoss method. Our reproducibility codebase can be found in the following repository [1]

[1] https://github.com/timur26/ICLR_Reproducibility_Challenge_Autoloss

2 BACKGROUND

To assess the reproducibility of Xu et al. (2018) and confirm the stated effectiveness of *AutoLoss*, we implemented the proposed methods as well as baseline methods and applied them to the d-ary quadratic regression, the MLP classification and the image generation using GANs.

2.1 D-ARY REGRESSION

In the d-ary regression problem, a quadratic model is fit to a synthetic dataset generated by a linear model with Gaussian noise. Since the model is over expressive, it's prone to overfitting, resulting in small training loss but large validation loss l_1. An additional regularizer loss l_2 is used to overcome overfitting in two ways: a linear combination of $l_1 + \lambda l_2$; an alternate minimization between l_1 and λl_2, where λ is a hyperparameter.

2.2 MLP CLASSIFICATION

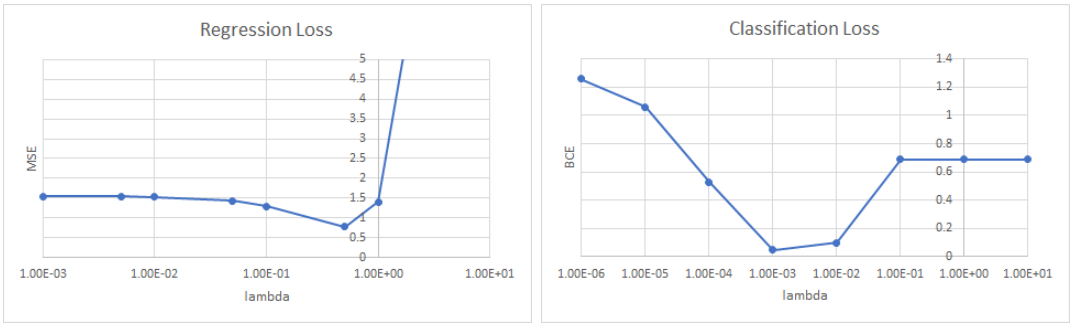
Similar to the regression, an expressive MLP model is learned for a binary classification task on a synthetic dataset. Given limited amount of training data, the MLP classifier is prone to overfitting. In addition to the binary cross entropy (BCE) loss l_1, a regularizer loss l_2 is incorporated in a linear combination $l_1 + \lambda l_2$ or through alternate minimization between l_1 and λl_2.

2.3 GAN IMAGE GENERATION

The objective function of GANs forms a minimax game between a generator (G) and a discriminator (D). It is generally hard to conduct joint optimization for both G and D simultaneously. As an alternative, seeking a saddle point by alternative minimization has been a common practice. While recent works from Radford et al. (2015); Arjovsky et al. (2017) have shown the importance of schedules between G update and D update, how to construct a optimal scheduling remains an open question.

3 JOINT MINIMIZATION BASELINES

We start with the joint minimization for the regression and classification problems. As the model is over expressive, a proper regularizer is needed to prevent serious overfit. Figure 1 shows the dense grid search of λ with respect to validation loss. Training details are discussed in the following subsections.



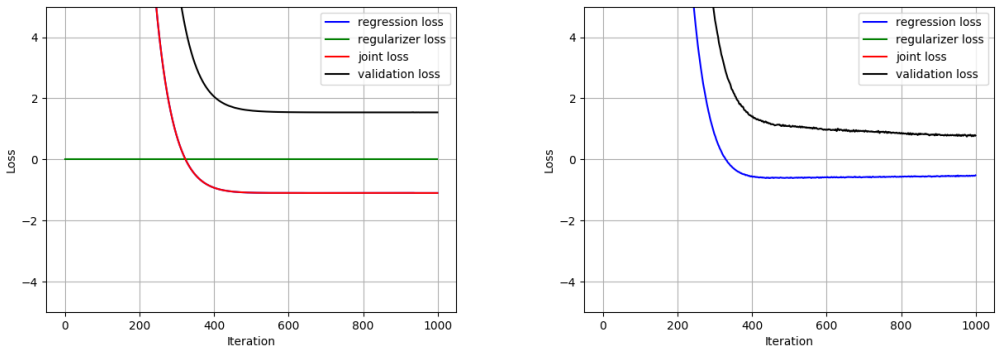
(a) regression (b) classification

Figure 1: Grid search for regularizer parameter with joint minimization

3.1 QUADRATIC REGRESSION

Figure 2 shows the joint minimization of the quadratic regressor for a synthetic dataset. The input dimension is 32, and the size of training set and validation set are 2000 and 5000 respectively. The MSE of the gaussian noise is substracted from the training loss l_1, as suggested in Xu et al. (2018).

As we can see, when the regularizer is negligible, the over expressive model seriously overfits the small training dataset, resulting in a negative regression loss (as it is subtracted by noise) but high validation loss. By contrast, adding a regularizer with a proper hyperparameter λ drastically reduces overfitting.

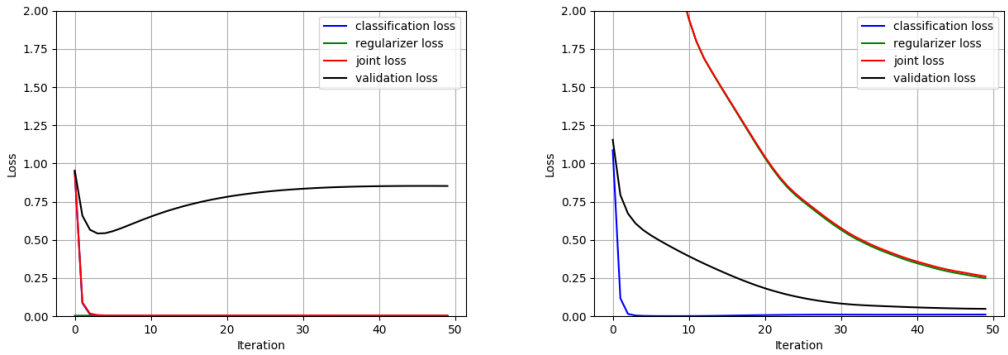


(a) $\lambda = 1e-10$ (b) $\lambda = 5e-1$

Figure 2: Joint minimization for regression with different weights of regularizer.

3.2 MLP Classification

Similar to the observations above, Figure 3 shows the training and validation losses for the classification problem. The model with good regularization demonstrates much smaller validation loss as compared to its counterpart without regularization.



(a) $\lambda = 1e-10$ (b) $\lambda = 5e-1$

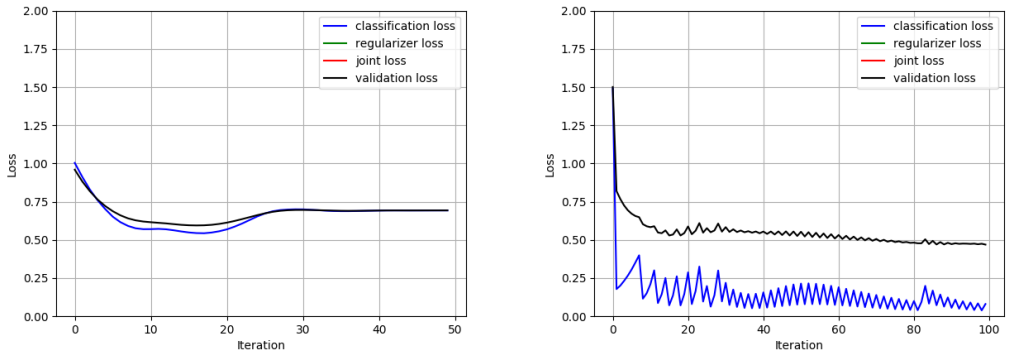
Figure 3: Joint minimization for classfication with different weights of regularizer.

4 Hand-crafted Schedules

As an alternative to joint optimization, updating models with respect to different objective terms alternatively is also a widely used approach in various problems. We next replicate a hand-crafted schedule, S1, introduced in as another baseline method for the classification problem.

For this schedule, the state of the training process is defined as $s = (l_1^{val} - l_1^{train})/l_1^{train}$, which indicates the degree of consistency between the loss on the training and the validation datasets. When s goes above a certain threshold th, the training switches to the minimization of the regularization loss l_2, otherwise it continues with the task objective l_1.

Figure 4 shows a comparison between the results of joint minimization and the use of the fine-tuned schedule, given a rather large $\lambda = 0.1$. Both plots display the validation loss (black) and the training loss (blue). As we can see, the alternate scheduling results in a validation loss of 0.47, which is significantly lower than the loss obtained using joint minimization with the same λ. The hand-crafted optimization schedule also outperforms the joint minimization with a small λ illustrated in Figure 3. This clearly demonstrates the effectiveness of the alternating schedule, that balances the training loss and validation loss to mitigate overfitting.



(a) Joint minimization

(b) Hand-scheduled alternative minimization

Figure 4: Comparison between joint minimization and hand-scheduled alternate minimization for classfication ($\lambda = 1e-1$)

5 AUTOLOSS SCHEDULES

As proposed in Xu et al. (2018), AutoLoss is a meta-learning framework that learns a schedule for alternate optimization from reward signals of a training process. We replicate their experiments in two tasks, classification and GAN, where the learned schedule demonstrates good performance.

Figure 5 shows the results of alternate optimization for MLP classification using two random schedules defined by the trained AutoLoss scheduler. Given a large λ, AutoLoss leads to highly competitive results compared to joint minimization and hand-crafted schedules with similar values of λ, revealing the potential of AutoLoss in discovering good schedules for classification problems.

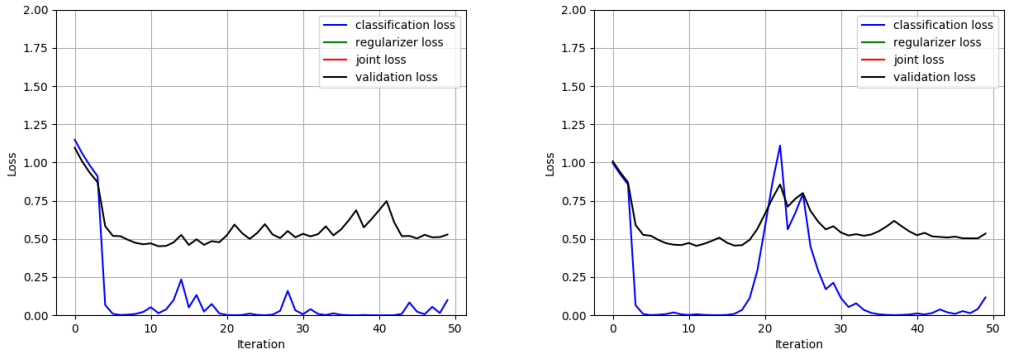
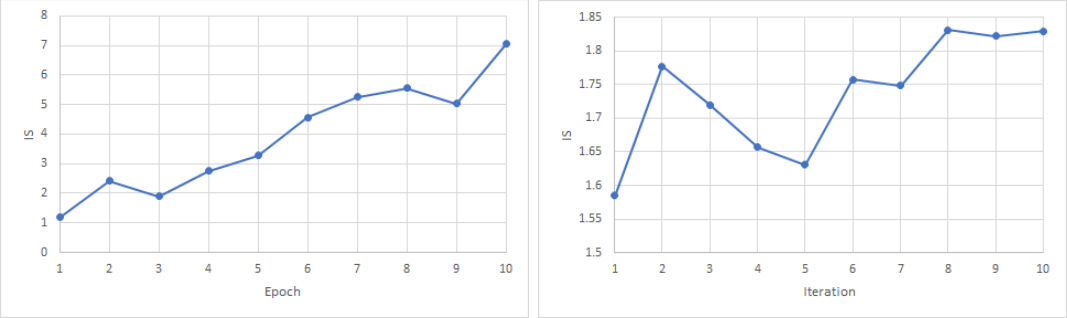


Figure 5: Meta-learned schedules for alternate minimization for classfication ($\lambda = 1e-1$)

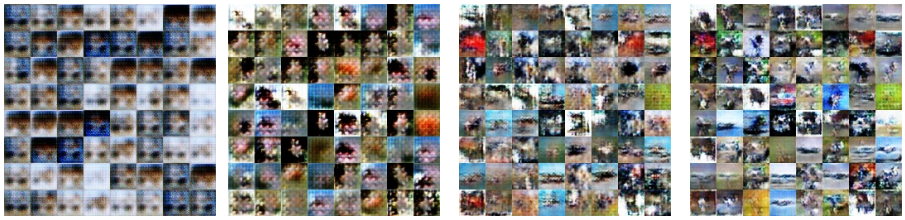
Figure 5 shows the learning curve of the Inception Score for a standard DCGAN scheduled by AutoLoss controller. The Inception Score keeps growing for Mnist, which shows the effectiveness of the implemented AutoLoss controller. While the Inception Score for Cifar10 increases rather slowly, the quality of generated images improve over the trainiing process, as shown in Figure 7.

A particular detail about our GAN autoloss implementation is that we reset the training of controller if we notice that the generator G loss diverges from the start. We check the divergence condition by checking the loss of generator after just 100 batch iterations of controller training process. If the loss is too high, we restart the controller training. This procedure does not add a significant overhead.



(a) Mnist (b) Cifar10

Figure 6: Evolution of DCGAN's Inception Score with AutoLoss



(a) Epoch: 0 (b) Epoch: 4 (c) Epoch: 8 (d) Epoch: 14

Figure 7: Evolution of DCGAN's Image Output with AutoLoss

6 Discussion

We observed quite a few noteworthy observations while training the AutoLoss schedule controller.

1. High Variance - If we do not update the initial parameters of the controller and keep them the same throughout the iterations, the accuracy of the output varies widely between 30% to 80%. Further investigation revealed that normalizing the features (keeping them at same scale in magnitude), provides some stability to the variance of the output which then ranges from 72% to 78%.

2. The Learning Rate Counter Intuition - The paper claims that AutoLoss works irrespective of the lambda parameter value of the l1 regularizer. But this is clearly not the case, as drastically increasing or decreasing lambda will lead to sub-optimal solutions. We do support the theory that given a sub-optimal value of lambda, AutoLoss can find a 'better schedule' than joint optimization of the 2 losses as well as 'manual scheduling'. (See Fig. 5)

3. Learning Rate Scheduling - After reviewing literature surrounding optimization, we realized that ML algorithms follow a simulated annealing scheduling. Initially, when the controller is training at the start, it is usually allowed to explore more (by providing a higher LR) and later as it converges to a good landscape, the LR is decreased for better convergence. The constant C in Algorithm 1 follows a similar behaviour. A higher value of C encourages exploration, whereas a lower value of C encourages convergence to a local minimum. We thus believe that the constant C should first start with a high value and gradually decrease in order to guide the controller a difficult landscape (non-convex in the case of the classification problem).

4. Increased Data Partitioning - In order to train the AutoLoss controller, the initial dataset must be split into 5 parts: training and validation datasets for both the task model and controller training, and a fifth partition to assess the task model after guided training of the controller. As a result, the

data dedicated to controller training and validation could instead be used to further train the task model which would also potentially result in greater accuracy. This observation does not seem to be explicitly stated in the paper under review.

7 CONCLUSION

In this project, we replicate experiments in three different tasks reported in Xu et al. (2018) namely regression, classification and image generation. Our empirical experiments on baseline methods show results similar to those presented in the paper. In addition to these baselines, we implement the manual scheduling in case of regression and classification as well as allow for the sampling of different GAN architectures as mentioned in appendix A.5 of Xu et al. (2018). Later, we report scenarios where we demonstrate the effectiveness of the proposed AutoLoss method for learning a good schedule in challenging tasks like classification where the loss landscape is non-convex as well as in GANs where manual scheduling is not easy to configure. Despite its working, we find the results are not very robust in terms of hyperparameters and problem settings. Prominent challenges faced include finding the right set of hyperparameters for the controller, particularly for the regression and classification problems.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Haowen Xu, Hao Zhang, Zhiting Hu, Xiaodan Liang, Ruslan Salakhutdinov, and Eric Xing. Autoloss: Learning discrete schedules for alternate optimization. *arXiv preprint arXiv:1810.02442*, 2018.