REPRODUCIBILITY REPORT —— UNDERSTANDING AND ROBUSTIFYING DIFFERENTIABLE ARCHITECTURE SEARCH

Wei Lou(wl4u19), Wenbin Su(ws4n19), Wentao Zheng(wz10y19)
MSc. Aritificial Intelligence
Code and report:https://github.com/Lewislou/COMP6248-Reproducability-Challenge

1 Introduction

1.1 NAS and DARTS

Neural Architecture Search (NAS) is a technology for automatically designing neural networks. Zoph & Le (2017) and Esteban Real & Le (2019) believe that NAS can be classified into 3 parts which includes search space, search strategy and performance evaluation. Nevertheless, this algorithms have the problem of massive calculation. One solution is to make the discrete optimization problem continuous. Liu et al. (2018) proposed an algorithm called Differentiable Architecture Search (DARTS), which converts the network structure search into a continuous space optimization problem, and uses the gradient descent method to solve the network structure α and the weight W.

$$\bar{o}^{(ij)}(x) = \sum_{o \in O} \frac{exp(\alpha_o^{(i,j)})}{\sum_{o' \in O} exp(\alpha_{o'}^{(i,j)})} o(x)$$

$$min_\alpha \quad L_{val}(w^*(\alpha), \alpha) \qquad s.t. \quad w^*(\alpha) = argmin_w \quad L_{train}(w, \alpha)$$

1.2 Problem of DARTS and improvement

Although the performance of DARTS is better than other algorithms, it is not robust in a few specific spaces. Therefore, the authors propose 2 hypotheses about this phenomenon. First one is bi-level optimization problem could lead to higher test loss. Authors observe the validation loss in 50 iterations and they find these errors decreased and convergent, which demonstrates a successful optimization. The next hypothesis is sharp local minima. For verifying this hypothesis, authors compute validation loss in full Hessian with small batches. The results show the reason of high loss is related to dominant eigenvalue increase. In addition, when the dominant eigenvalue is larger, it would lead to weaker generalization ability and affect pruning step because the curvature of loss function would be larger as dominant eigenvalue is larger.

As a result, authors propose 3 methods for solving the large dominant eigenvalue problem. The first method is early stopping for avoiding high curvature and the other two focus on regularization. Authors uses CutOut (CO, DeVries & Taylor (2017)), DropPath (DP, Barret Zoph & Le (2018)) for data augmentation and increasing L2 regularization. Moreover, the results denote early stopping can work together with these 2 regularization and its test error is lowest.

1.3 Project reproduces

Due to the limitation of GPU resources and time, this project focuses on reproducing the results of above three improved methods for using the CIFAR-10 dataset on s1 space and propose the final result is consistent with the result of the thesis.

2 Experimental Methodology

2.1 The early stop for Dominant Eigenvalue

In order to avoid the high curvature, the author has applied the early stop, which is to monitor the maximum eigenvalue in the Hesse matrix of validation loss function in the process of structure search and stop the search if it is about to reach a large value, to the original DARTS. In this paper, Zela et al. (2020) has implemented it by a heuristic algorithm. Using $\theta^\alpha_{max}(i)$ to represent the value of λ^α_{max}, which is the largest eigenvalue of the Hessian matrix of validation loss, smoothed over $k = 5$ epochs around i, it will calculate the $\theta^\alpha_{max}(i-k)/\theta^\alpha_{max}(i)$ once at every k epochs. If it is smaller than 0.75, the training will stop updating and return to the architecture at the $i - k$ epoch.

2.2 Data augmentation

Different from the idea of early stop, Zela et al. (2020) also argues that the modification of inner objective of training loss can effectively flatten the output parameters because the internal optimization process determines the external parameters in the bi-level optimization in this experiment. Therefore, author used data augmentation algorithms, which are Cutout and Scheduled-DropPath to achieve this idea as it can randomly add some noise to the dataset, which can avoid the overfitting to some extend for improving generation ability.
For the Cutout, it is a method that train a model on randomly cropped images. This method helps the model to improve the cognitive ability of the picture, because the covered image will make the model better focus on the its local features, which contribute to improving the generation ability. For the other method used in this paper, Scheduled-DropPath is to discard the path in the cell with a certain probability. Compared to the original DropPath, the path drop probability in the Scheduled-DropPath increases linearly in the training process due to the likelihood of overfitting increases with the number of training epoch.

2.3 Increased L2 regularization

The other regularization used to modify the inner objective of train loss is to increase the L2 regularization. L2 regularization is an approach to decrease the weight by adding a regularization term. In this paper, Zela et al. (2020) increases the L2 regularization in order to make the weight much smaller. This is able to improve the generation ability as the high weight will make the model sensitive to the data disturbance such as noise and lead to a poor generation ability.

$$L(x,y) = \sum_{i=1}^{n}(y_i - h_\theta(x_i))^2 + \lambda \sum_{i=1}^{n}(\theta_i^2)$$

3 Implementation

3.1 The main changes in the Robust DART

The code implementation is mostly based on the original DART implementation Liu et al. (2018) and the Robust DART implementation Zela et al. (2020). The whole experiment process of this paper is basically the same as the original DART. The search network chooses useful operations from search space by computing the bi-level function using validation loss and training loss. Then the architecture is retrained using CIFAR-10 dataset to get the final testing error and accuracy. The main changes are:1. Adding more search spaces for comparison. 2. Adding the Hessian matrix and dominant eigenvalue calculation with early stopping. 3. Adding cutout augmentation, L2 regularization, Drop path methods

3.2 Evaluation of Robust DART

From the paper, the authors show the failure of the original DART in 4 different spaces. In order to show the better performance of the Robust DART, the architectures trained by these 2 methods need to be compared. After the network search, the operations that chosen from the model can be printed out. And the final error can be compared. So 8 experiments for these 2 methods on 4 spaces need to be performed.

Besides, the 3 methods used in this paper need to be evaluated. In this project, the final metrics for evaluating these parameters will be the classification error for CIFAR-10 dataset. The experiment parameters can be concluded as 13 different experiments show in table.1: Wd means the weight decay value and Dp means the drop path probability.

Table 1: Experiment configurations

Model	Robust DART											
Space	S1									S2	S3	S4
Wd	0	0.0003	0.0009	0.0027	0.0081	0.0243	0.0003					
Dp	0						0.2	0.4	0.6	0		

4 Experiment and Analysis

All the experiments are performed on the ECS Yann GPU server with 4*1080ti graphic processing units and 16 * Intel 7400 CPU. However, due to the high usage of this server for so many ECS students, we decide to only use one single GPU with maximum 10000MB memory for each training experiment.

Table 2: Final accuracy affected by DropPath prob

Search Space	Drop Prob.	Valid acc.	Test acc.	λ^{α}_{max}
s1	0.0	98.320000%	87.536000%	0.5211677640742106
	0.2	83.436000%	84.232000%	0.25210391241453245
	0.4	75.548000%	81.400000%	0.16456613116105925
	0.6	62.736000%	75.704000%	0.11512142667802039

4.1 The architecture search result

Ideally, the search result of the DART methods needs to be printed out. However, we found that the DART has high GPU memory requirement problem about 12000MB for batch size 64 and the dynamic memory increasing problem. After a few epochs, GPU memory usage will increase, so when other user uses that GPU, our training process will report out of memory problem. Besides, when set the batch size as 32, the training time for each epoch will be around 2 hours. If we train the DART model for 50 epochs like the epoch number from the paper, it would spend about 5 days to perform an experiment in original DART which is not suitable for us.
So for comparison of the search result, we will only show the result of the robust dart model which cost about 12 hours for each experiment. And compare to the results state in the paper. It's clear that the results of Robust DART show more reasonable operations in network architecture. For example, the original DART uses all skip connection operations in S1, but the robust DART model used max_pooling which is more meaningful. Also in S4, when adding noise operation which is a harmful operation, the original DART will choose noise, but robust DART avoids to do it.
Based on the search architecture comparison of these 2 methods, we can say that this method used in the paper is actually more robust and well-performed than the original version in different search spaces.

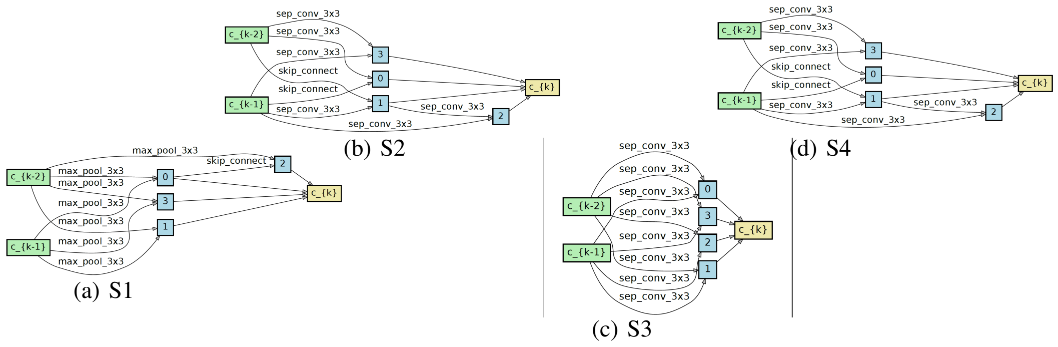


Figure 1: The arichitecture search result of 4 spaces

4.2 The effect of regularization methods

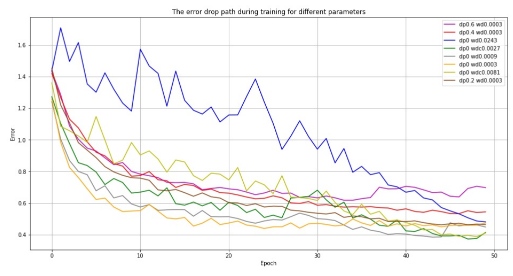
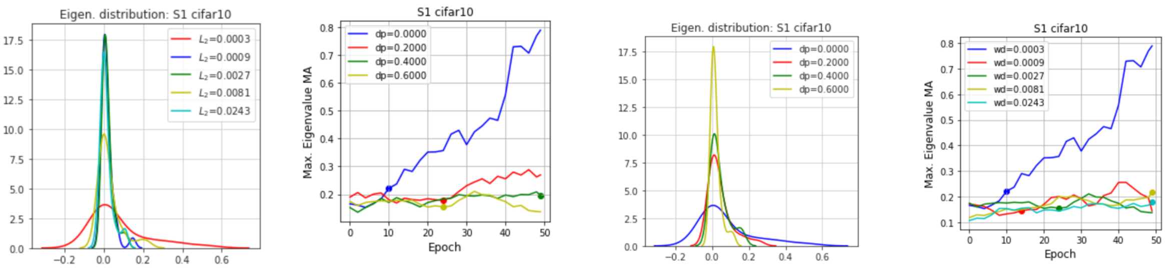


Figure 2: The error drop paths

4.2.1 Weight Decay and Early Stopping

The Fig.1 shows the relationship of L2 value and the eigenvalue distribution, it seems that by using L2 regularization, it helps for controlling the eigenvalue and keeping it to a small value which



(a) Fig.1 (b) Fig.2 (c) Fig.3 (d) Fig.4

Figure 3: The relationship between eigenvalues and wd,dp

explains the reason that the model can be more robust. Also, Fig.2 shows the effect of early stopping, the blob points are the early stopping point calculated by the Robust DART model which helps to control the maximum eigenvalues. This figure also shows larger L2 value would help to control the eigenvalue better. The figure not only shows the good performance of L2 normalization which lower down the testing error but also shows that though larger L2 value can bring better result, too larger value can cause a bad result.

4.2.2 Drop path and Early Stopping

The Fig.3 demonstrate that larger drop path probability value will help to control the distribution of the eigenvalue, for example, when dp equals to 0.6, most of the eigenvalues are around 0.0, but if dp equals to 0.0, lots of eigenvalues are very large. Also, Fig.4 shows that with drop path regularization and early stopping, it' very successful to not trap into sharp minimal. Table.2 expresses the final accuracy increase by using good drop path probabilities. And compare to the result of the original DART paper, the final accuracy is slightly better than their result.

4.2.3 Efficiency

The training time for each epoch on original DART method is about 1.5 hour if using batch_size 64. However, the Robust DART will only use 12 minutes for each epoch. The reasons why this method can be faster are using early stopping and L2 regularization. The L2 regularization can make the model find the optimal easier which reduces the training time, and the early stopping can help the model avoid from overfitting also reduces the training time.

4.2.4 Other problems

There is a memory increasing problem in Robust DART as well. For example, on epoch 0, the memory usage is 2679 MB, but on Epoch1, the memory becomes 4735MB, and the end of the training process, the overall memory usage will be 5000MB. This might because of the GPU usage during the validation process and the hessian matrix eigenvalues storage for early stopping. This problem causes out of memory error several times. So we decide to add memory automatic clean function in the code and transfer the validation process to CPU. Besides, the original Robust DART project did not design the resume mechanism, so each time the training process failed we have to run it again from the beginning. So we added a resume function in the code so that we can resume training from the interrupted epoch.

5 Conclusion

This report shows some of the core ideas of this paper, like maximum eigenvalue early stopping, cutout, L2 regularization and drop path methods. And evaluated them in different search space and CIFAR-10 dataset. These methods achieve great performance as demonstrated in the paper, compare to the original DART method, it achieves better accuracy and much more robustness. Also, the Robust DART method training is more efficient and cost much less time than DART. However, due to the limitation of computational resource and time, we only reproduce the s1 s4 space and tuned some of the parameters. Also, it's a pity that we can not run the original DART project on the GPU server.

References

Jonathon Shlens Barret Zoph, Vijay Vasudevan and Quoc V. Le. Learning transferable architectures for scalable image recognition. *In Conference on Computer Vision and Pattern Recognition*, 2018.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Yanping Huang Esteban Real, Alok Aggarwal and Quoc V. Le. Aging evolution for image classifier architecture search. *In AAAI*, 2019.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018. URL `https://github.com/quark0/darts/tree/f276dd346a09ae3160f8e3aca5c7b193fda1da37`.

Arber Zela, Thomas Elsken, Tonmoy Saikia, Yassine Marrakchi, Thomas Brox, and Frank Hutter. Understanding and robustifying differentiable architecture search. 2020. URL `https://openreview.net/forum?id=H1gDNyrKDS`.

Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. *In International Conference on Learning Representations*, 2017.