Final report

Prerequisite program for Studienarbeit/Projektarbeit

Suraj Bhardwaj

University of Siegen

Faculty IV: School of Science and Technology

Department of Computer Science

Visual Computing Group & Computer Vision Group

November 17, 2022

Confirmation

I hereby confirm that this report is entirely my own work and that I have not used any additional assistance or resources other than indicated. All quotations, paraphrases, information and ideas that have been taken from other sources (including the Internet as well as other electronic sources) and other persons' work have been cited appropriately and provided with the corresponding bibliographical references. The same is true of all drawings, sketches, pictures and other illustrations that appear in the text. I am aware that the neglect to indicate the used sources is considered as fraud and plagiarism in which case sanctions are imposed that can lead to the suspension or permanent expulsion of students in serious cases.

Siegen, 17. Nov. 2022
Place, Date

Suraj Bhardwaj
Signature

Contents

Chapter 1

Introduction

In this study, the data augmentation method "AUGMIX" [1] is taken into consideration in order to assess its robustness to common corruptions and perturbations [2]. [3] coined the term ResNet, which stands for residual networks. The ResNet18 architecture consists of 72 layers with 18 deep layers [4]. ConvNeXT is a pure convolutional model invented by [5] using Vision Transformers design choices. Three datasets CIFAR-10 [6], CIFAR-10-C, and CIFAR-10-P [2] were used to carry out the experiments for this study. The evaluation was based on the evaluation criteria clean/test error, mean corruption error (mCE), and mean flip probability (mFP), whose mathematical formulations are described in [1] and [7].

1.1 Objectives

The main tasks assigned for this study are:

1. The addition of neural networks "Resnet18 and ConvNeXt-tiny" to the augmix code (not-pretrained and pre-trained).

2. Training of both neural networks on CIFAR-10 and Evaluation on CIFAR-10, CIFAR-10-C, and CIFAR-10-P datasets with:

(a) AdamW optimizer and cosine annealing learning rate scheduler.

(b) SGD optimizer and lambda learning rate scheduler.

3. Hyperparameter tuning of convnext-tiny model to improve it's performance.

Chapter 2

Analysis of Experiments

2.1 Analysis of Experiment 1

In the first experiment, the Augmix method is trained on the CIFAR-10 dataset using the Resnet-18 and ConvNeXt-tiny models (with pretrained and not-pretrained weights) respectively). Both networks begin with a learning rate of 0.1 and decay using a cosine learning rate function, as described by [8]. Before AugMix augmentations, the CIFAR-10 train dataset was pre-processed. The stochastic gradient descent (SGD) is optimized over 100 training epochs using Nesterov momentum [1].

2.2 Analysis of Experiment 2

The second experiment is split into two sections. In the first step, the Augmix method with both models is trained on the CIFAR-10 dataset using the AdamW optimizer, cosine annealing learning rate scheduler and default settings of [1]. The models are only evaluated on CIFAR-10-C dataset to get Mean Corruption Error(mCE). The results of Test error and mCE presented in Table 1 shows that the performance of the Resnet18 model (with pretrained weights), ConvNeXt-tiny (with both pretrained and not-pretrained weights) models are very poor,

Table 1: Results of AdamW optimizer without hyperparameter tuning on both models.

Models	Pretrained	Clean/Test Error	mCE
Resnet18	False	24.66	30.28
Resnet18	True	90.00	90.00
Convnext-tiny	False	90.00	90.00
Convnext-tiny	True	90.00	90.00

Table 2: Results of AdamW and Cosine AnnealingLR with lr=0.001 and weight-decay=0.0001.

Models	Pretrained	Clean/Test Error	mCE
Resnet18	False	12.52	18.104
Resnet18	True	11.06	16.705
Convnext-tiny	False	18.30	23.611
Convnext-tiny	True	8.38	15.033

prompting the search for accurate hyperparameter tuning.

The hyperparameters for each of the models are discovered in the second part of this experiment through a literature review and model experimentation. The learning rate, weight decay, batch size, and number of epochs were the hyperparameters under consideration. The performance of the models significantly improved when learning rate decayed from 0.1 to 0.001 and weight decayed from 0.0005 to 0.0001. Table 2 displays the results of this experiment. To examine the effect of hyperparameter value selection on the ConvNeXt-tiny pretrained model, the weight decay value is increased from 0.0001 to 0.01. This configuration produced a Test error of 8.55 and a mean corruption error of 15.127. Compared to Table 2, both errors have increased slightly, so I chose hyperparameter values in the opposite direction. In other words, a decrease in the learning rate, weight decay, and an increase in the number of epochs enable the ConvNeXt-tiny model to outperform previous results. However, in order to reduce the computation time, a correct combination is still desirable.

The actual hyperparameter values for the ConvNeXt-tiny model are taken from [5] and are as follows: learning rate = 5e-5 and weight decay = 1e-8. In

addition, the epochs are tuned based on the train loss curve's convergence, with a value of 94 chosen for the ConvNeXt-tiny model with pretrained weights. Though the hyperparameter tuning is best for the ConvNeXt-tiny pretrained model, for comparison, all models are trained on CIFAR-10 with this setting and then evaluated on CIFAR-10, CIFAR-10-C, and CIFAR-10-P datasets.

2.3 Results

2.3.1 Results of experiment 1

The results of Experiment 1 are shown in Table 3. According to Table 3, the Resnet18 not-pretrained model performed well among others based on the experiment settings. It can be seen here that, despite being a very complex model, the convnext-tiny model does require hyperparameter tuning and a standard optimisation procedure to achieve better results. Figure 1 depicts a plot of Test Error v/s. Epochs for all models under investigation in experiment 1.

2.3.2 Results of experiment 2

The final results of Experiment 2 are shown in Table 4. It shows that, based on the experiment settings, the Convnext-tiny pre-trained model outperformed others on the clean and mean corruption errors. It means that the convnext-tiny model, when fine-tuned on hyperparameter values, produces better robustness results for corruptions, or that the convnext-tiny model is resistant to corruptions. On the contrary, its performance on perturbations is very poor, indicating that

Table 3: Results of SGD + LLR on both models.

Models	Pretrained	Clean/Test Error	mCE	mFP
Resnet18	False	11.15	16.837	0.01577
Resnet18	True	12.28	18.076	0.10658
Convnext-tiny	False	39.72	44.887	0.11928
Convnext-tiny	True	30.08	36.901	0.21571

Figure 1: SGD: Clean/Test Error results

there is a trade-off between clean error, mCE, and mFP. Similar trade-off between clean accuracy, robustness, and uncertainty is also demonstrated by [9]. These results, however, are limited to the final setting of experiment 2. As shown in Table 2, separate hyperparameter tuning for each model produced a different result. Figure 2 depicts a plot of Clean Error v/s Epochs for all models under investigation in experiment 2.

Table 4: Results of AdamW and Cosine Annealing LR with epochs=94, lr=5e-5 and weight-decay=1e-8

Models	Pretrained	Clean/Test Error	mCE	mFP
Resnet18	False	21.74	27.969	0.01577
Resnet18	True	13.26	19.253	0.10658
Convnext-tiny	False	27.49	32.422	0.11928
Convnext-tiny	True	5.80	12.226	0.21571

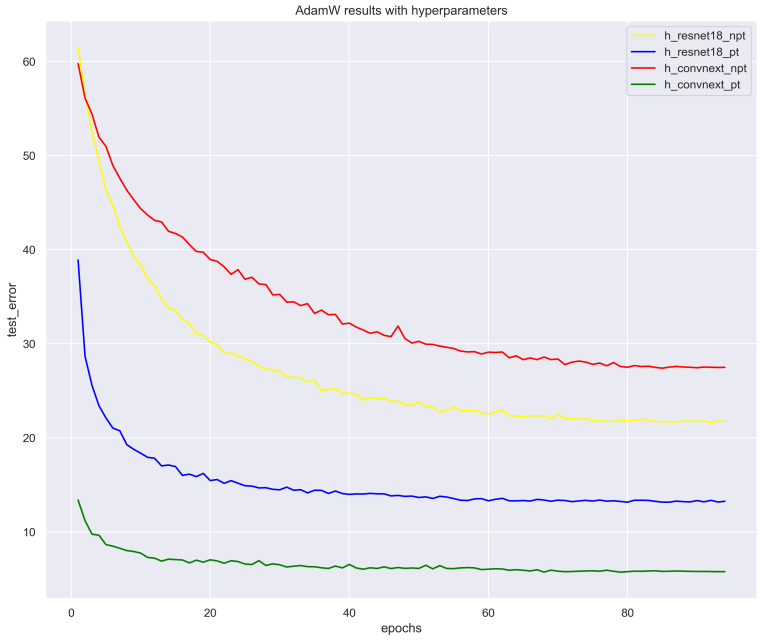

Figure 2: AdamW: results on Test error

Chapter 3

Conclusion

The Resnet18 (not-pretrained) model trained on CIFAR-10 and optimized using SGD and Lambda learning rate scheduler performed best on the CIFAR-10 and CIFAR-10-C datasets, with a clean error of 11.15 and mCE of 16.837, respectively. When trained using AdamW optimizer and CosineAnnealingLR Scheduler with hyperparameter values of learning rate=0.001, weight decay=0.0001, and epochs=100, Resnet18(pretrained) model outperforms these results with a Test error of 11.06 and mCE of 16.705. This leads to the conclusion that when properly tuned, the AdamW optimizer in conjunction with the CosineAnnealingLR scheduler outperforms the Stochastic Gradient Descent optimizer in conjunction with the LambdaLR scheduler. However, the trade-off between corruption and perturbation errors is still there and is evident from the outputs of mean Flip Probability values. Irrespective of the methods of model optimization used in this study, the mean Flip probability value is the same for all model comparisons when compared as Resnet18 (SGD) v/s Resnet18(AdamW). It leads to the conclusion that optimization techniques do not affect perturbations, but this is not true for corruptions as illustrated in Figure 3.

Furthermore, the ConvNeXt-tiny model (with pretrained weights) outperforms other models only on Clean and mean Corruption errors. For the mean Flip probability, the results show that a model's performance on mean Corrup-

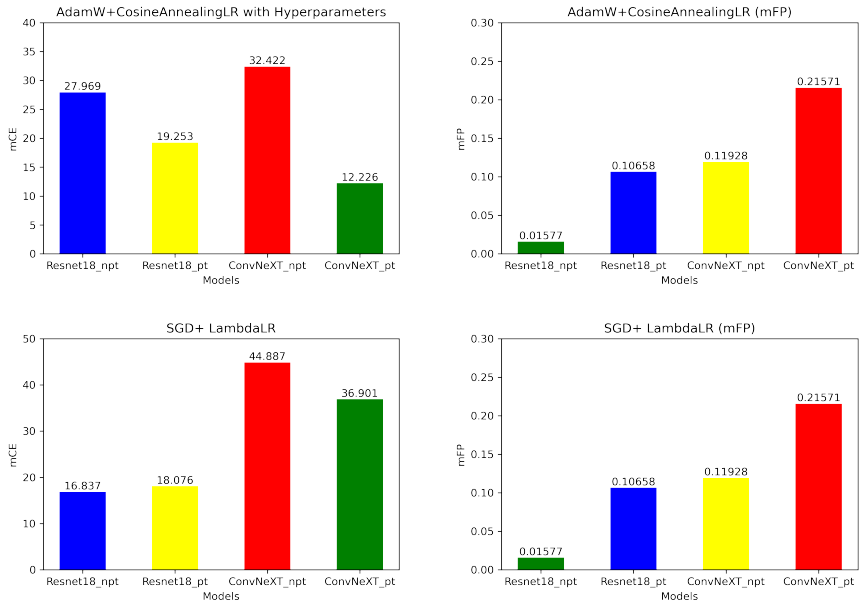

Figure 3: Comparison of results on all models

tion error is poorer than its performance on mean Flip probability, indicating a trade-off between mean corruption error and mean-flip probability. On the ConvNeXt tiny model with pretrained weights, a separate experiment was run with the final settings of experiment 2, except for a larger batch size of 256. This experiment produced relatively good results in a shorter amount of time, with clean error = 5.94 and mean-Corruption error = 12.324. This shows a trade off between efficiency of the model and compute time.

Bibliography

[1] D. Hendrycks*, N. Mu*, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple method to improve robustness and uncertainty under data shift," in *International Conference on Learning Representations*, 2020.

[2] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *Proceedings of the International Conference on Learning Representations*, 2019.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[4] Y. Chandola, J. Virmani, H. Bhadauria, and P. Kumar, "Chapter 4 - end-to-end pre-trained cnn-based computer-aided classification system design for chest radiographs," in *Deep Learning for Chest Radiographs* (Y. Chandola, J. Virmani, H. Bhadauria, and P. Kumar, eds.), Primers in Biomedical Imaging Devices and Systems, pp. 117–140, Academic Press, 2021.

[5] Z. Liu, H. Mao, C. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," *CoRR*, vol. abs/2201.03545, 2022.

[6] A. Krizhevsky, G. Hinton, *et al.*, "Learning multiple layers of features from tiny images," 2009.

[7] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *CoRR*, vol. abs/1903.12261, 2019.

[8] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with restarts," *CoRR*, vol. abs/1608.03983, 2016.

[9] S. Chun, S. J. Oh, S. Yun, D. Han, J. Choe, and Y. Yoo, "An empirical evaluation on robustness and uncertainty of regularization methods," *arXiv preprint arXiv:2003.03879*, 2020.