Final report

Training program for Studienarbeit/Projektarbeit

Nishilkumar Balar

University of Siegen

Faculty IV: School of Science and Technology

Department of Computer Science

Visual Computing Group & Computer Vision Group

May 4, 2023

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, 04.05.2023

Place, Date

Signature

Contents

1 Introduction

Under identical training and test distribution, modern deep neural network generally achieve high accuracy. However, when test distribution differs in terms of distribution in comparison to training one, models performance in terms of accuracy significantly deteriorate. This happens possibly due to neural networks' tendency to memorize properties of the specific training pattern. In [1], AugMix, a data processing technique, is proposed which is easy to implement and less computational expensive. "AugMix significantly improves robustness and uncertainty measures on challenging image classification benchmarks, closing the gap between previous methods and the best possible performance in some cases by more than half." [1]

In this work, as a part of student training program, AugMix is used with ResNet18 [2] and ConvNeXt-T [3] networks to train on Cifar-10 [4] dataset. To check robustness and uncertainty estimate of trained network, evaluation is carried out on Cifar-10, Cifar-10-C and Cifar-10-P [5] dataset. In addition, different optimization approach such as AdamW [6] and Stochastic gradient descent (SGD) are used on both networks with ImageNet [7] dataset pretrained weights and without pretrained weights. Also, It is important to mention that cosineannealing learning rate scheduler is already implemented in AugMix code with lambda learning rate scheduler using *get-lr()* function. Therefore, the description of its influence is explicitly avoided in this report.

2 Experiments

In this chapter, experiment setup and its results are discussed in details. As described earlier, ResNet18 (RN) and ConvNext-Tiny (CNT) neural network architecture are used with both pretrained and not pretrained weights. As shown in [1], clean test error, mean corruption error (mCE) and mean Flip Probability (mFR) metrics are evaluated to check performance of implemented network with specific optimizer and learning rate.

2.1 Setup

The 8 different set of experiments are performed as shown in the table below. Here, for each training, number of epoch is taken as 100. While different learning rate (lr) is used to get desired accuracy during training and evaluation.

Experiment Setup				
Exp. No.	Architecture	Pretrained	Optimizer	Learning Rate
1	ResNet18	No	AdamW	0.025
2	ResNet18	No	SGD	0.025
3	ResNet18	Yes	AdamW	0.015
4	ResNet18	Yes	SGD	0.015
5	ConvNextTiny	No	AdamW	0.004
6	ConvNextTiny	No	SGD	0.004
7	ConvNextTiny	Yes	AdamW	0.003
8	ConvNextTiny	Yes	SGD	0.003

2.2 Result

In this section, obtained results of previously mentioned metrics are as shown in the table below.

Results of Experiment					
Arch.	Pretrained	Optimizer	Clean Error	mCE	mFR
RN	No	AdamW	12.68	18.225	0.027
RN	No	SGD	14.646	17.213	0.024
RN	Yes	AdamW	12.82	18.601	0.027
RN	Yes	SGD	10.97	16.623	0.024
CNT	No	AdamW	15.41	20.333	0.898
CNT	No	SGD	45.99	50.215	0.900
CNT	Yes	AdamW	11.62	18.529	0.026
CNT	Yes	SGD	7.34	13.716	0.019

2.3 Discussion of Results

The experiment table results clearly show that for the ConvNext-Tiny architecture, the network with pretrained weights performs better for both AdamW and SGD optimizers in comparison to the non-pretrained model. The possible reason for this could be that ImageNet pretrained weights might provide a good initial point to start training, which converges the model towards a better local optimum solution than the non-pretrained one.

For the non-pretrained model with the AdamW optimizer, the ConvNext-Tiny architecture performs as desired, but with the SGD optimizer, the resultant model acts as a random classifier. In this case, different hyperparameters such as learning rate, number of epochs, batch size, and weight decay can be tuned to get satisfactory results with the SGD optimizer. If a pretrained ConvNext-Tiny model is used, we can

conclude from the results that the SGD optimizer performs better than the AdamW optimizer. This could be because SGD is better at escaping local minima and finding the global optimum of the loss function.

However, there is a significant improvement in the accuracy of the model as well as generalization in terms of mCE and mFR for the pretrained ConvNext-Tiny architecture with SGD optimizer, possibly due to better starting weights that allow the network to learn meaningful feature representation and avoid overfitting.

In addition, for the ResNet18 architecture, both the pretrained and non-pretrained models perform almost identically on unseen perturbation or corruption of Cifar10-p and Cifar10-c, respectively. However, like in the ConvNext-Tiny model, the pretrained ResNet model with SGD optimizer performs superior, possibly for the same reasons.

When ResNet18 and ConvNext-Tiny are compared, they differ significantly in terms of the depth and complexity of the network architecture. ResNet18 is a much deeper and more complex network with more parameters, which allows it to capture more complex patterns in the data, but also requires more computation and memory. ConvNext-Tiny, on the other hand, is a much simpler and more lightweight architecture that is designed for use on low-power platforms but may not be able to capture as much complexity in the data as ResNet18.

Overall, the results show that in general, ResNet18 performs better during evaluation than the ConvNext-Tiny model. However, among all eight performed experiments, the pretrained ConvNext-Tiny model with the SGD optimizer has the lowest clean error, mCE, and mFR. There could be several reasons why pretraining ConvNext-Tiny with SGD optimizer led to significant improvements in accuracy and generalization in terms of mCE and mFR.

Firstly, pretraining with SGD optimizer allows the network to learn meaningful feature representations on a large dataset, which can then be fine-tuned on the target dataset. This can lead to better generalization performance because the network has already learned to recognize and classify a wide range of features and patterns in the data.

Secondly, the pretrained model may have been able to avoid overfitting on the target dataset because it had already learned to recognize and classify similar features in the pretraining dataset. This can help to prevent the model from becoming too specialized to the target dataset, which can limit its ability to generalize to new data.

Finally, pretraining with a large dataset can also help to regularize the network and prevent it from overfitting. This is because the network has to learn to recognize and classify a wide range of features and patterns in the data, which can help to prevent it from focusing too much on specific features or patterns that are only present in the target dataset.

Finally, pretraining with a large dataset can also help to regularize the network and prevent it from overfitting. This is because the network has to learn to recognize and classify a wide range of features and patterns in the data, which can help to prevent it from focusing too much on specific features or patterns that are only present in the target dataset.

3 Conclusion

A ResNet18 trained model is generally more robust during evaluation on unseen corruption and perturbations in a dataset than a ConvNext-Tiny trained model. However, a pre-trained ConvNext-Tiny model with the SGD optimizer outperforms significantly in comparison to ResNet18 and non-pretrained models. For Cifar datasets with or without any perturbation or corruption, the SGD optimizer gives a better model that generalizes well in comparison to the AdamW optimizer. Additionally, hyperparameters such as the number of epochs, learning rate, batch size, weight decay, etc., may also influence the training of the neural network to find the minimum of the loss function.

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] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[3] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," 2022.

[4] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research),"

[5] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," 2019.

[6] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.