Project Report for Imagenette Classification

Sishuo Chen 1700012765
School of Electronic Engineering and Computer Science, Peking University
chensishuo@pku.edu.cn

Abstract

This report is a description of my implementation for Imagenette classification, as a practice project for the Introduction to Artificial Intelligence course in Peking University, spring 2021. I train a Resnet18 model which reaches 92.48% test accuracy as my baseline. Furthermore, I discuss the effect of several factors including the depth of the model, the choice of training batch size and the effect of weight decay by conducting ablation study experiments. My source code has been uploaded along with this report.

1 Task Introduction

Imagenette[1] is a subset of 10 easily classified classes from Imagenet (Deng et al., 2009) (tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute). I use the full size verision of Imagenette [2] for all experiments in this report. The training set consists of 9469 pictures and the test set consists of 3925. The aim of this project is building a neural network for classifying Imagenette and the final test accuracy should be more than 92%.

2 Baseline Implementation

2.1 Experimental Setting

I implement my baseline model with Pytorch (Paszke et al., 2019) .

For model architecture, I choose Resnet18 , a 18-layer residual convolutional network (He et al., 2016) with about 45 MB of parameters.

I choose SGD as the training optimizer with the momentum set to 0.9 and the weight decay hyper parameter set to 5e-4. The learning rate is set to 0.1 during the first 150 epochs,then reduced to 0.01 during the next 100 epochs and 0.001 for the last 100 epochs(350 epochs in total). The batch size is set to 32. Random resize crop and random horizontal flip is used for data augmentation during training.

2.2 Main Results

2.2.1 Accuracy and Loss Curve

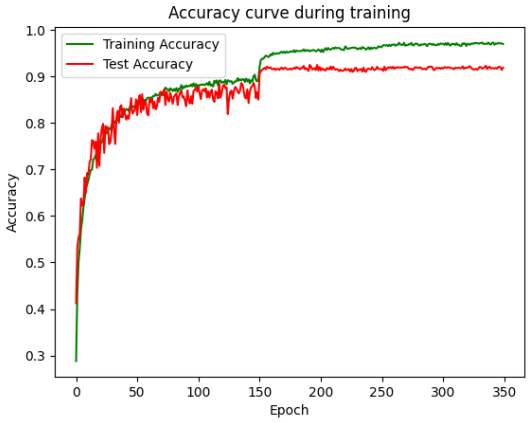

Figure 1: Accuray Curve

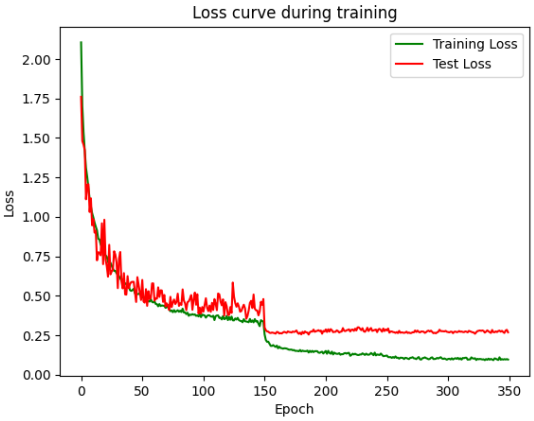

Figure 2: Loss Curve

[1] https://github.com/fastai/imagenette/

[2] https://s3.amazonaws.com/fast-ai-imageclas/imagenette2.tgz

The model reaches 92.48% accuracy on the test set. I test the model after each training epoch and report the best test accuracy which is reached after epoch 192. Figure 1 and 2 show the change of loss and accuracy during training. The images show that both training and test error is sharply reduced after the learning rate reduction in epoch 150. After reaching best test accuracy at around epoch 200, the training error continues to descend while the test error remains almost constant, which seems to suffer from over-fitting.

2.2.2 Time Cost

The model is trained on 4 NVIDIA Titan RTX GPUs. The training process costs about 5.5 hours. (350 epochs in total, batch size=32)

3 Ablation Study & Analysis

I discuss the effect of several factors including the depth of the model, the choice of the optimizer and training batch size by conducting ablation study experiments.

3.1 Model Depth

Imagenette is a small dataset and a 18-layer residual network seems deep enough to fit it. To check it, I explore the effect of model depth by increasing the model depth. I test 18-layer, 34-layer, 50-layer, 101-layer and 152-layer Resnets, while the training hyper parameters and hardware setting remain the same as baseline.

Layers	Accuracy %	Size	Time Cost
18	92.48	45MB	5.5h
34	92.64	83MB	14h
50	**93.50**	98MB	30h
101	92.33	170MB	35h
152	92.41	230MB	42h

Table 1: The effect of model depth

Results in table 1 show that the 50-layer model reached the highest test accuracy at 93.50%. When I increase the depth to 101 and 152, the test performances begins to drop. Considering that the training set consists of less than 10k images and pretraining is not allowed in this project, too large model size risks overfitting. It seems that a 50-layer model is most suitable to fit it.

3.2 Batch Size

Modern deep neural network training is typically based on mini-batch stochastic gradient optimization. Small batch training has been shown to provide improved generalization performance and allows a significantly smaller memory footprint ; large batch training is faster to converge but tends to converge to sharp minimizers of the training and testing functions and leads to poor generalization(Keskar et al., 2017; Masters and Luschi, 2018). Resnet uses batch normalization (Ioffe and Szegedy, 2015), which has been used extensively in deep learning to achieve faster training process and better resulting models. Lian and Liu (2019) shows that BN's solution is sensitive to the mini-batch size and it suffers from a too small batch size. So the choice of training batch size is a open problem without a fixed answer. Empirically, the best classification performance has been consistently obtained for mini-batch sizes between m=2 and m=32 (Masters and Luschi, 2018).

Batch Size	Test Accuracy %
8	89.68
16	92.41
32	**92.48**
64	91.77
128	90.06

Table 2: The effect of batch size

To check the effect of training batch size, I train the Resnet18 model with m = 8,16,32,64 and 128, and the results are shown in table 2. The best test performance is reached with m = 32. Whether I use larger or smaller batch size, the test performances will fall.

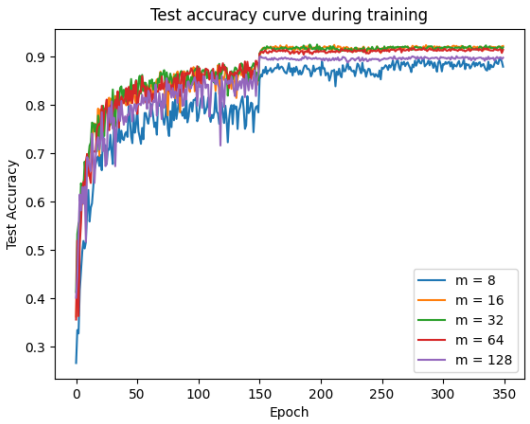

Figure 3: Test accuracy curves under different batch sizes

Figure 3 shows the test accuracy curves under different training batch sizes. It's evident that both

λ	**Test Accuracy %**
0	90.68
1e-4	91.97
5e-4	92.48
1e-3	92.82
5e-3	**93.02**
1e-2	91.67

Table 3: The effect of weight decay

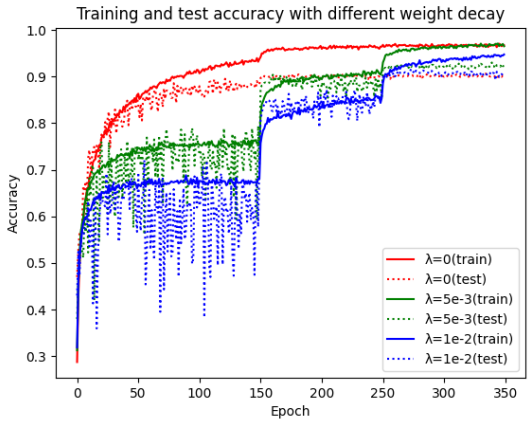

Figure 4: Test accuracy curves under different batch sizes

too large and too small batch sizes lead to unstable training progress and poor final performance. m = 32 seems to be a suitable choice.

3.3 Weight Decay

Weight decay is a regularization technique by adding a small penalty, usually the L2 norm of the model weights, to the loss function. It provides an approach to reduce the over-fitting of a deep learning neural network model on the training data and improve the performance of the model on new data, such as the holdout test set. The L2 regularized cost function can be written as $C = C_0 + \frac{\lambda}{2n}\sum_w w^2$, in which C_0 refers to the original loss function and n refers to batch size. The hyper parameter λ controls the size of L2 penalty. In all experiments mentioned before, λ is set to 5e-4.

To explore the effect of λ, I test a series of choices of λ in comparison with the baseline. Table 3 shows the results. When I increase λ from 0 to 5e-3, the test performance keeps growing, which reaches 93.02% when $\lambda = 5e-3$, 0.54% higher than the baseline. But the performance begins to fall with a larger λ.

To better understand the effect of weight decay, I plot the training and test accuracy curves under $\lambda = 0, 5e-3, 1e-2$ in figure 4.

When there is no weight decay ($\lambda = 0$, the red curves in the figure), the model

When λ is too large ($\lambda = 1e-2$, the blue curves in the figure), the model shows under-fitting: both training and test accuracy is lower than the model trained with a proper λ ($\lambda = 5e-2$, the green curves in the figure).

4 Conclusion

In this project, I practice training neural networks for image classification on the Imagenette dataset. The Resnet18 model reaches 92.48% accuracy and Resnet50 model reaches 93.50% under my default hyper parameter settings. In addition, I explore the effect of model depth, training batch size and weight decay on model training by conducting exhaustive control experiments and discuss the results and corresponding insights in this report.

References

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2017. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Xiangru Lian and Ji Liu. 2019. Revisit batch normalization: New understanding and refinement via composition optimization. In *The 22nd International*

Conference on Artificial Intelligence and Statistics, AISTATS 2019, 16-18 April 2019, Naha, Okinawa, Japan, volume 89 of *Proceedings of Machine Learning Research*, pages 3254–3263. PMLR.

Dominic Masters and Carlo Luschi. 2018. Revisiting small batch training for deep neural networks. *CoRR*, abs/1804.07612.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.