CSE 455 Homework 5

Yoshihiro Kumazawa

August 27, 2020

1 Installing PyTorch

Done!

2 Find the best network

2.1 Training a classifier using only one fully connected Layer

See Figure 1a. We can say that the the model successfully trained since the loss is decreasing throughout the training process and there is a healthy gap between the training accuracy and testing accuracy.

2.2 Training a classifier using multiple fully connected Layers

See Figure 1b. The training is not successful because the testing accuracy plateaus whereas the training keeps increasing.

2.2.1 Question

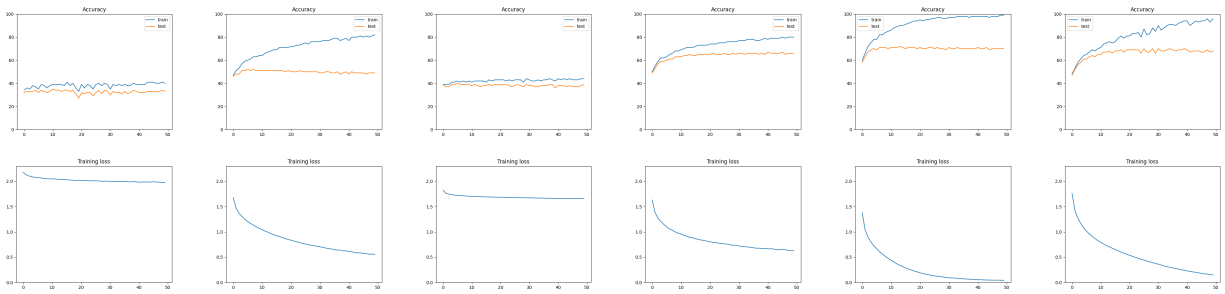
See Figure 1c. The model accuracy is significantly worse than the previous model. This is because the model is expressively limited since it has less non-linearity. The model can actually become just as good as LazyNet since without activations, the forward pass is just a couple of matrix multiplications, which is nothing but a single matrix multiplication by the composed matrices. But somehow, by separating the weight update process in back propagation, it achieves a slightly higher accuracy than LazyNet.

2.3 Training a classifier using convolutions

Our CoolNet is based on LeNet [2]. In addition, batch normalization [1] is performed after each convolutional layer and linear layer except the final layer in order to get the network to converge faster. As a result, the network got higher accuracy then the previous models. See Figure 1d for the training result.

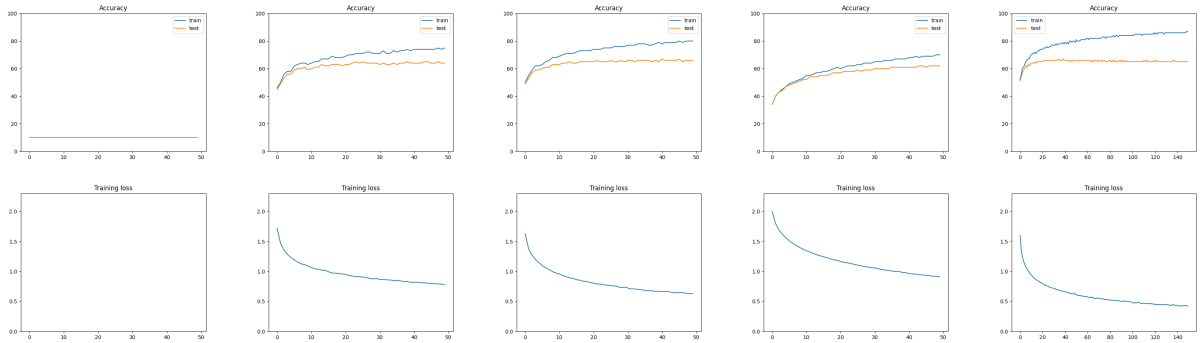
2.3.1 Question

We tried batch sizes of 4, 32 and 256. See Figure 1d, Figure 1e, and Figure 1f respectively. The loss curves account for the difference. Batch size 4 is so small to compute gradients accurately that the loss does not decrease sufficiently. On the other hand, when the batch



(a) (b) (c) (d) (e) (f)

Figure 1: Training results of LazyNet, BoringNet and CoolNet: (a) LazyNet, (b) BoringNet, (c) BoringNet without activations, (d) CoolNet with batch size 4, (e) CoolNet with batch size 32, (f) CoolNet with batch size 256.



(a) (b) (c) (d) (e)

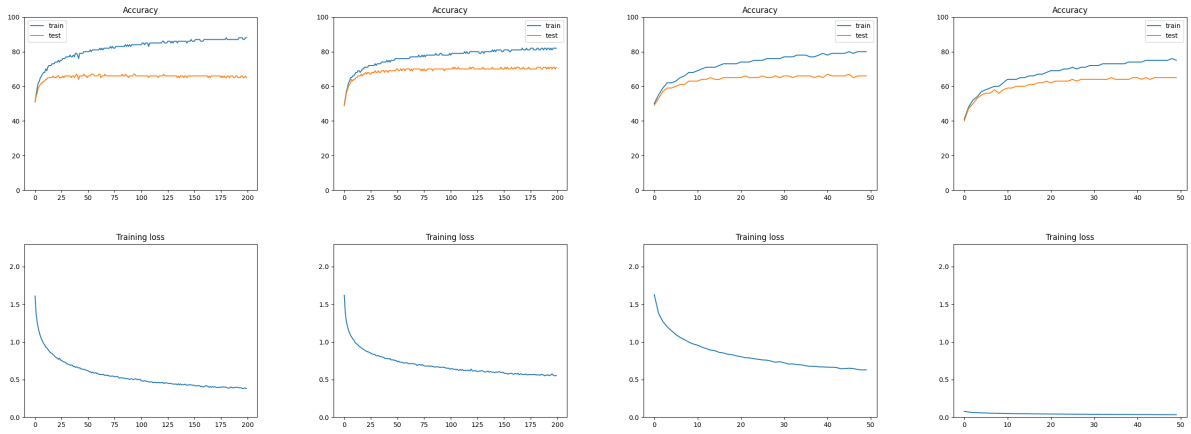
Figure 2: Training results of CoolNet. (a) lr=10, (b) lr=0.1, (c) lr=0.01 (same as 1d), (d) lr=0.0001, (e) epoch=150. For (a), the training loss was as high as 4.1 throughout the training process, which is so high that it does not show up in the figure.

size is 256, the total iteration number is much smaller and the training loss does not saturate. Among these 3 models, the model trained with batch size 32 performs best.

3 How does learning rate work?

See Figure 2a through 2d. The model with learning rate 0.01 (Figure 2c) performs best. since its validation accuracy is the highest. Its training loss is the lowest too.

Figure 2e shows the training result of the model trained for 150 epochs. The training accuracy is better than the 50-epoch model (Figure 2c) but the testing accuracy is just as good. This is because the model is overfitting at an early stage of training, in which case training longer will not help.



(a) (b) (c) (d)

Figure 3: Training results of CoolNet. (a) epoch=200, (b) epoch=200 with horizontal flipping augmentation, (c) Cross entropy loss, (c) Mean squared error loss.

4 Data Augmentation

See Figure 3b and Figure 3a. The testing accuracy is significantly better when horizontal augmentation is added although the training loss is higher. That means it successfully alleviated the problem of overfitting.

5 Change the loss function

See Figure 3c and Figure 3d. Note that the loss values are not comparable. The model trained with MSE took more time to converge than the one with CE. One possible reason is that for the correct class, the gradient $-2(1-p)$ of the MSE loss is smaller in modulus than the gradient $-1/p$ of the CE loss, which leads to slower training.

References

[1] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[2] Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.