Low-data Image Classification Using Transfer Learning

Yifan Bai

November 8, 2020

1 Methodology

Rarely does one train a neural network, usually CNN, from scratch for image classification. Pretrained models are used instead First, we usually cannot get enough labeled data to train such a model; secondly, even if we have sufficient data, it would be time-consuming.

1.1 Models

In this project, we leverage transfer learning to overcome limitations posed by extremely small dataset. Specifically, we consider three popular pretrained models which we aim to compare and explore their own advantages and disadvantages:
AlexNet [4]. Classic GCN model consists of 5 conv layers and 3 fully-connected layers.
ResNet [2]. Most SOTA methods in image classification leverage the residual connections. ResNet's mechanism is simple, i.e. shortcutting the features of bottom layer, usually the input, to the deeper layer. Oftentimes, the model consists of tens of neural layers, which causes the information of initial input features cannot be conveyed to the very deep layers. Nontheless, by connecting the shallow layer with the deeper one, the model actually learns the residual between the output and the input, which is much easier to be optimized.
DenseNet [3]. DenseNet can be regarded as an extension of ResNet, as it adds much more shortcuts in the model. However for DenseNet the residual connections are implemented by channel-wise concatenation, rather than element-wise addition used in ResNet.

1.2 Data Augmentation

Many SOTA approaches use data augmentation to boost data size. We use the method from SimCLR [1], where original image is randomly cropped and then colour-distorted.

2 Implementation and Results

The project is implemented using the framework with Torch 1.3.1, and the pretrained models are provided by its build-in tools [1]. For all three models, the learning-rate was set to 0.001 and optimized with SGD with momentum

[1]https://pytorch.org/docs/1.3.1/torchvision/models.html

The dataset has three different classes with each one containing 10 samples. We used 5-folds cross validation, and $Train : Test : Valid = 6 : 2 : 2$. Specially, we randomly split the data of each class into five folds and then combine the data, such that the proportion of each class in each fold is balanced. At the end, we report the results of 20 epochs.

2.1 Classification Results

The results are shown in Table 1. Few key observations: (1) Although AlexNet had the least loss on the testing set, its accuracy was lower than that of DenseNet. Also, the performance of ResNet was outperformed by DenseNet. (2) Data augmentation consistently improved the performance of all three models, matching our expectations.

Table 1: Classification results (5-folds cross validation).

Models	Average Loss	Accuracy
AlexNet [4]	**0.162**	0.900
ResNet [2]	0.420	0.867
DenseNet [3]	0.385	**0.933**
AlexNet w/o DA	0.342	0.867
ResNet w/o DA	0.433	0.867
DenseNet w/o DA	0.427	0.900

2.2 Comparing the Losses of Three Models

Figure 1 depicts the losses in one fold without data augmentation that helps to better understanding. We can see that AlexNet [4] had the most stable loss curve with least fluctuations, however, it did show significantly more overfitting. The curves of the other two fluctuated much more, but as suggested from the previous section, they still managed to achieve decent results and there are less overfitting. This in general implies that ResNet [2] and DenseNet [3] could have better generalization when use pretrained models without data augmentation. Figure 2 shows the results with data augmentation, and we can see that overfitting have been reduced for all three models. Particularly, DenseNet [3] and ResNet [2] narrowed down the loss deficit to that of AlexNet [4]. Indeed, as mentioned in the previous section, data augmentation is beneficial in performance.

3 Conclusion

This project aims to study using deep learning to solve an image classification problem with extremely low amount of data. In order to do so, we used transfer learning that leverages both data augmentation and pre-trained networks. While each model has its own advantages and disadvantages, we managed to achieve decent results. From accuracy standpoint, DenseNet [3] was the best performer, while AlexNet [4] has the least loss. We also confirmed that data augmentation improves performances.

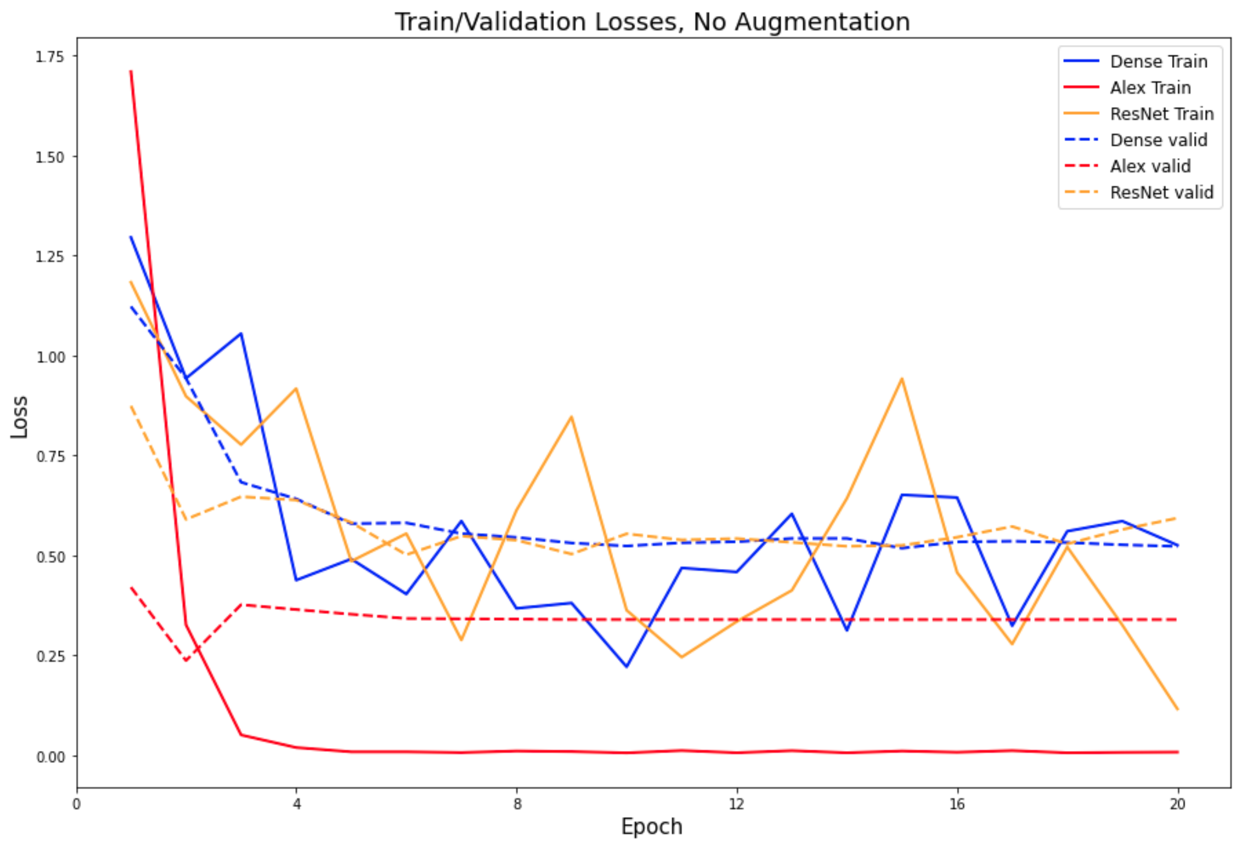

Figure 1: Train/Validation Loss without Data Augmentation

H]

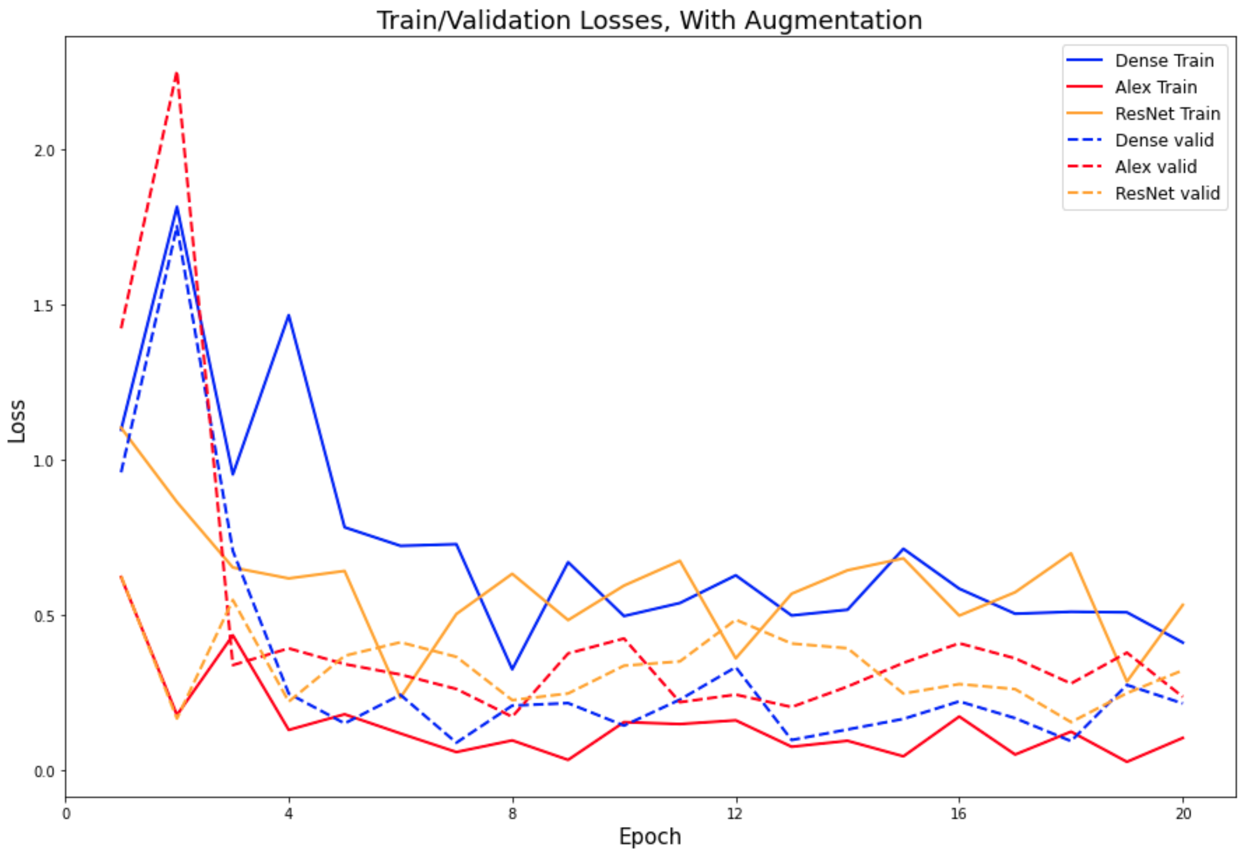

Figure 2: Train/Validation Loss with Data Augmentation

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations, 2020.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[3] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[4] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014.