Lab4: Diabetic Retinopathy Detection

311551147 高振群

April 2023

1 Introduction

Deep learning is a field that addresses problems using deep neural networks, where the depth (i.e., the number of layers) of the network directly influences its power. In theory, deeper networks are expected to perform better. Given a network N with n layers and another network M with m layers (where m > n), let's define the task set T as the set of tasks that the shallower network N can solve, and task set $T^{'}$ as the set of tasks that the deeper network M can solve. We can construct M by copying N and adding $m - n$ identity mappings. This construction ensures that M performs identically to N on any existing task. Therefore, we have $T \subseteq T^{'}$, which means the deeper network is at least as good as, if not better than, the shallower network according to theory.

However, in practice, it has been observed that deeper networks may not always outperform shallower networks, contrary to theoretical expectations. Researchers often attribute this to the problem of vanishing gradients. The vanishing gradient problem occurs when the gradient decreases exponentially during backward propagation, hindering the update of network parameters. Deeper neural networks are more susceptible to this problem because the backward propagation takes a longer path in the computational graph compared to shallower networks. As a result, experimental results sometimes contradict theoretical predictions.

In 2015, the deep residual network (ResNet) was introduced as an attempt to address the vanishing gradient problem. The shortcut connections in ResNet prevent the propagated gradient from becoming overly small during backward propagation, mitigating the vanishing gradient issue. In this lab, we will be using a pre-trained ResNet model for image classification in diabetic retinopathy detection. By leveraging the advantages of deep networks while avoiding the vanishing gradient problem, ResNet has achieved an accuracy of approximately 83 percent.

2 Experiment setups

2.1 The details of your model (ResNet)

In this lab, the ResNet18 and ResNet50 models are utilized, implemented using the PyTorch package, and following the original ResNet paper for consistency. In section 1, the advantages of ResNet18 were discussed. Now, the focus shifts to the bottleneck design employed in ResNet50 (refer to Figure 1). The bottleneck design reduces the number of channels in the intermediate layers, resulting in fewer parameters and lower computational cost. This makes it suitable for deep networks, which often face challenges with high computational cost due to a large number of parameters.

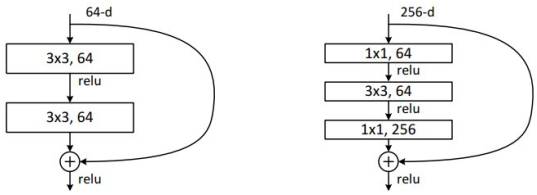

Figure 1: The design of the skip layer in ResNet. Left: the normal design used in shallower ResNet. Right: the bottleneck design used in deeper ResNet

2.2 The details of your Dataloader

The implementation of the dataloader can be found in the "dataloader.py" file. For both training and testing, the batch size was set to 32 for ResNet18 and 16 for ResNet50. The dataloader reads the images and labels from the dataset using the CSV file. In the training stage, it also applies data augmentation techniques to augment the data and enhance the model's ability to generalize.

2.3 Describing your evaluation through the confusion matrix

The confusion matrices depicting the results are displayed from Fig. 2 to 5. The correct predictions are represented along the diagonal of the matrix. Notably, the training set and test set exhibit an imbalance in class distribution, with class 0 having significantly more samples compared to other classes. This class imbalance poses challenges in accurately classifying the other classes in the dataset resulting in the weird confusion matrices of .

Remarkably, the pretrained models do not exhibit such drastic differences in results compared to the un-pretrained group. Another point to observe is that even when the predicted result is incorrect, the distance between the prediction and label is relatively small. This could be attributed to the fact that the disease

progresses gradually. For example, the features extracted from a mild patient should be close to the features extracted from a moderate or no DR patient instead of proliferate DR patient.

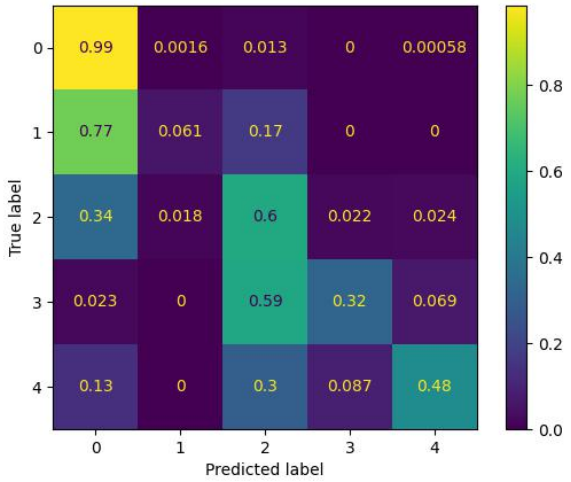

Figure 2: Confusion matrix of pre-trained ResNet18.

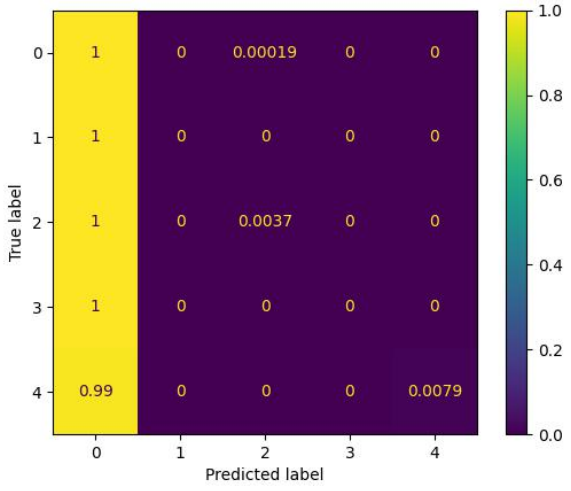

Figure 3: Confusion matrix of ResNet18.

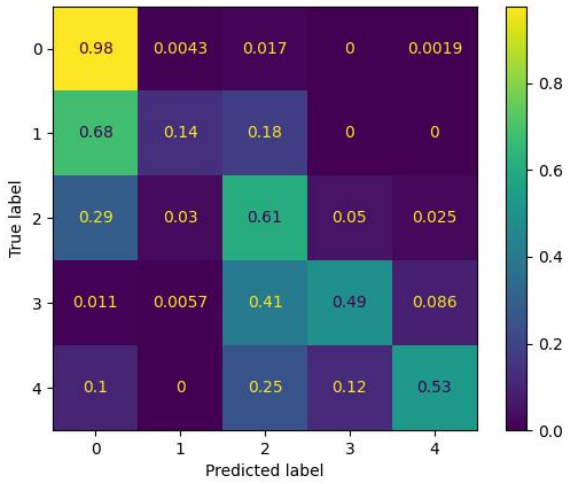

Figure 4: Confusion matrix of pre-trained ResNet50.

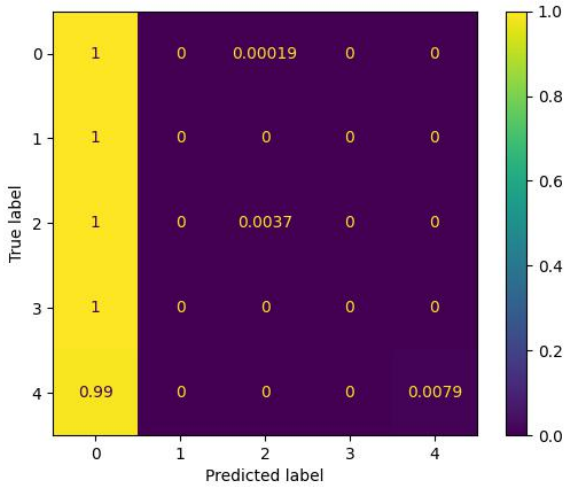

Figure 5: Confusion matrix of pre-trained ResNet50.

3 Data preprocessing

3.1 How you preprocessed your data?

The data is initially clipped into a square image with dimensions of (m, m), where m is the minimum of the height and width of the original image. The clipped image is then resized to $(512, 512)$. During the training phase, horizontal flipping is applied with a 50% probability, and random rotation in the range of -10 to 10 degrees is also applied.

3.2 What makes your method special?

To prevent distortion during resizing, the image is first clipped into a square shape. Additionally, horizontal flipping and random rotation are applied as data augmentation techniques during training. These techniques help generate more training data from the existing images and prevent overfitting, ultimately enhancing the model's ability to generalize to unseen data.

4 Experimental results

4.1 The highest testing accuracy

The highest test accuracy is shown in Table 1.

model	pre-trained	accuracy
ResNet18	yes	0.8355
ResNet18	no	0.7340
ResNet50	yes	0.8415
ResNet50	no	0.7332

Table 1: Test accuracy of ResNet.

4.1.1 Screenshot

The screenshot are shown from Fig. 6 to 9.

Figure 6: Accuracy of pre-trained ResNet18.

Figure 7: Accuracy of ResNet18.

Figure 8: Accuracy of pre-trained ResNet50.

Figure 9: Accuracy of ResNet50.

4.2 Comparison figures

The comparison figures are shown in Fig. 10 and 11. It is obvious that the pretrained model outperform its counterpart. It's probably due to the fact that pretrained model already has the capability of image feature extraction.

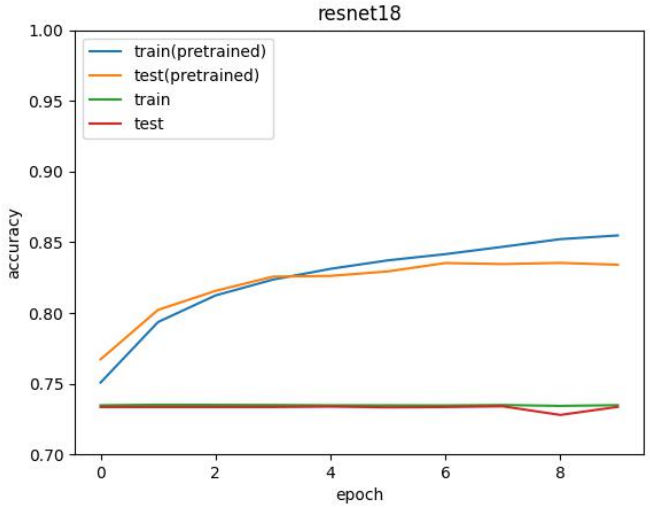

Figure 10: The comparison figure of ResNet18.

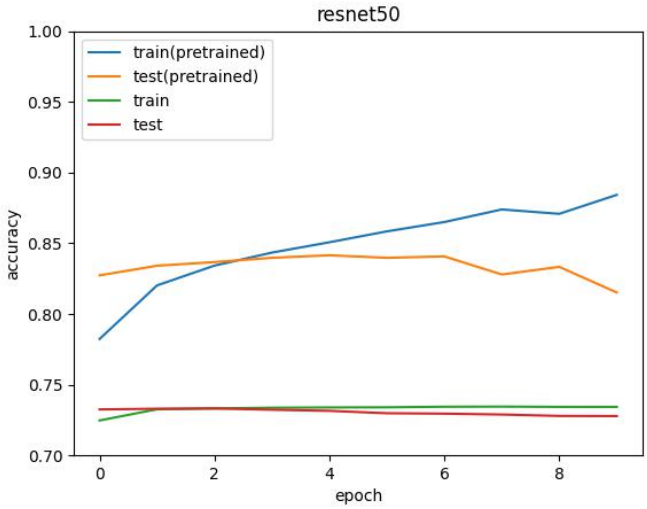

Figure 11: The comparison figure of ResNet50.

5 Discussion

In this lab, ResNet18 and ResNet50 were implemented and trained. The results demonstrate that ResNet50 performs at least as well, if not better, in terms of accuracy compared to ResNet18. This observation suggests that the skip layers in ResNet help mitigate the issue of gradient vanishing, as deeper networks tend to perform better according to theoretical considerations. However, as shown in Fig. 11, ResNet50 is more susceptible to overfitting, which is intuitive due to its higher capacity compared to ResNet18. To address this, regularization techniques such as dropout or weight decay could be applied to prevent overfitting in ResNet50.