National Tsing Hua University
Fall 2023 11210IPT 553000
Deep Learning in Biomedical Optical Imaging
Homework 4

CHUN-HAO LI[1]

1Ultrafast Photonics Lab, Institute of Photonics Technologies, Tsing Hua University.

Student ID:112066513

1. *Task A: Modal Selection*

For this part, the two pre-trained models are: AlexNet and GoogLeNet. The visualized structure of the two models is shown in Fig.1. and Fig.2.
First of all, for AlexNet, is the first neural network that introduce Convolution Neural Network [1], it is an 8 layers structure model, with 5 layers of convolution layer and 3 layers of fully connected layer. It contains 61.1 million parameters, and the Top-1 accuracy 56.522%, the Top-5 79.066% in the other hand. The feature of AlexNet is that it stacks convolutional layers to extract the features of the images and in some degree reduces overfitting. And it uses ReLU as activation function reduces the calculation and fasten the training speed. It applies local response normalization to suppress the response of ReLU also increase the generalization. Also, AlexNet add dropout mechanism in the fully connected layer to reduce the complexity of neurons in order to prevent overfitting. Last but not least, it uses multiple GPUs and fasten the training time.

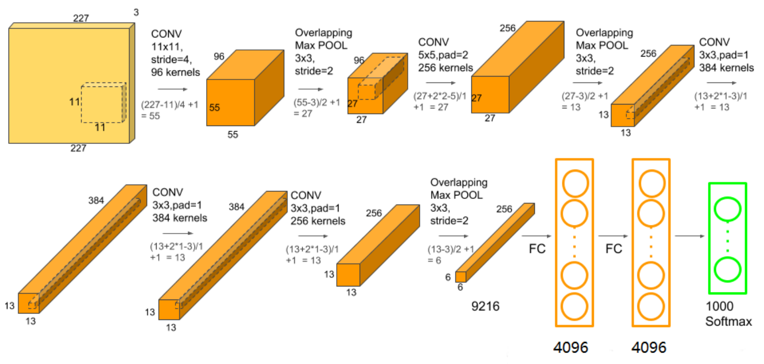
CONV 11x11, stride=4, 96 kernels
(227-11)/4 +1 = 55
Overlapping Max POOL 3x3, stride=2
(55-3)/2 +1 = 27
CONV 5x5,pad=2 256 kernels
(27+2*2-5)/1 +1 = 27
Overlapping Max POOL 3x3, stride=2
(27-3)/2 +1 = 13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
CONV 3x3,pad=1 384 kernels
(13+2*1-3)/1 +1 = 13
CONV 3x3,pad=1 256 kernels
(13+2*1-3)/1 +1 = 13
Overlapping Max POOL 3x3, stride=2
(13-3)/2 +1 = 6
9216
FC
FC
4096
4096
1000 Softmax

Fig.1. Structure of AlexNet [3]

And next is the GoogLeNet, the specific structure of its basic is the "Inception" structure. With 22 layers for the first version of the model and some optimization for the newer models. It has only 6.6 million parameters, and the Top-1 accuracy is 69.788%, the Top-5 89.53% in the other hand. The main characteristics is that it uses Inception structure rather than merely convolution and activation. Inception network is the concatenation of Inception module, with additional convolution layer to change the width and height of the modle, the strides are all 1 throughout the structure. C.Szegedy et el present significant work in "Going deeper with convolutions" [2]. We can consider the two side fully connected layer structure as assisted classifier, they play a role of regularizer, and they can pass gradient more efficient.

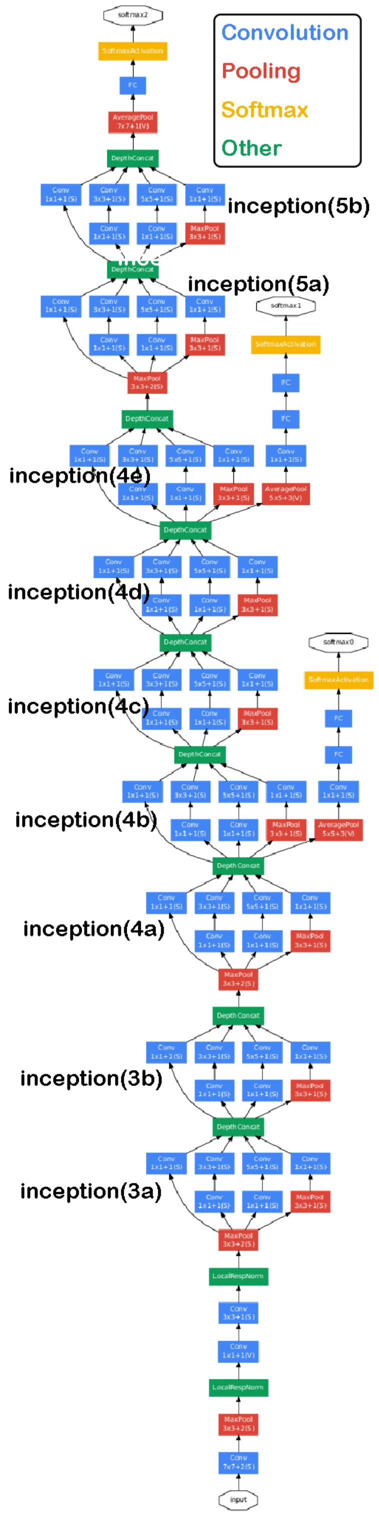

Fig.2. Structure of GoogLeNet

2. *Task B: Fine-tuning the ConvNet*

After fine-tuning the two models for the purpose of our 2-chest x-ray dataset, and running both training as well as testing, here is the results for them:

Table 1. Training Results for AlexNet and GoogLeNet

Model	Best Train accuracy	Best Train Loss	Best Validation accuracy	Best Validation Loss	Training Time
AlexNet	97.06%	0.0811	96.75%	0.0759	1:33
GoogLeNet	100%	0.0018	99.5%	0.0127	4:06

As we can see from Table 1. and compare the overall training performances from Fig.3, Fig.4, It seems that the training performance of GoogLeNet is better than AlexNet, as it can avhieve higher training and validation accuracy, on the other hand, lower training and validation loss. As I test the two models with same condition, the testing accuracy of AlexNet is 72.5%, and 74% for GoogLeNet. But although GoogLeNet has less parameters, the convolution layers are more than AlexNet, I assume this is the main reason for training time of GoogLeNet is almost three times longer than AlexNet.

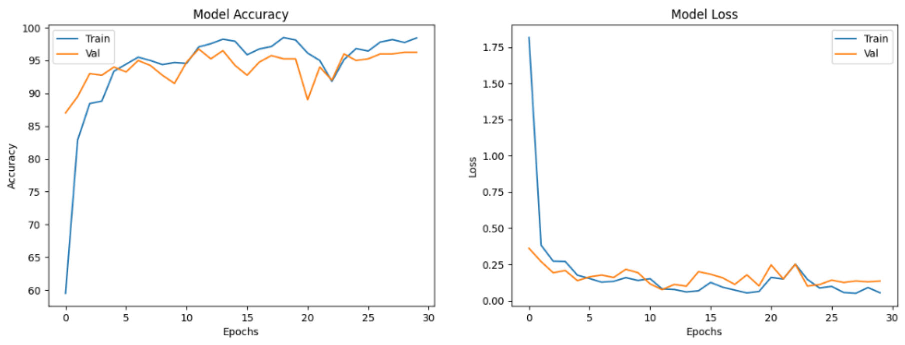

Fig.3. Overall Training Performance of AlexNet

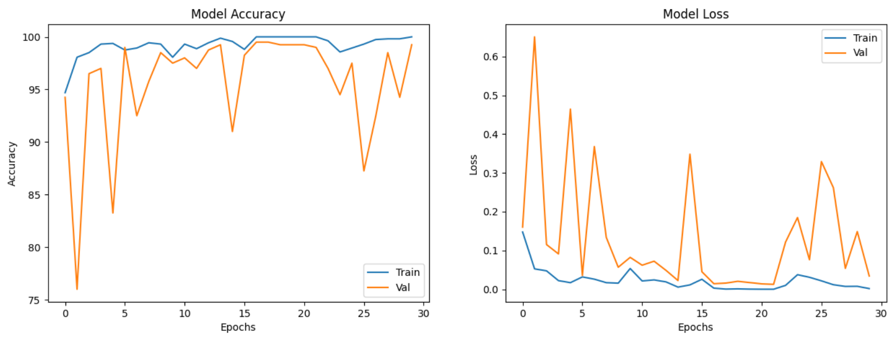

Fig.4. Overall Training Performance of GoogLeNet

3. *Task C: ConvNet as Fixed Feature Extractor*

After transforming the AlexNet model into fixed extractor, I come across an error code which says "`element 0 of tensors does not require grad and does not have a grad_fn`", and this prevent me from continue the training process. When I look into the source code of AlexNet, the "element 0" refers to "Dropout" function, so I assume the error is because there aren't any parameters which can be calculated gradient descent for the backward propagation. Then I search for ways to let this backward propagation can keep going, the first method I found is shown below:

```
loss.requires_grad = True
```

But after applying this code, another problem shows up, which is: both train and validation accuracy are 0, with both train loss and validation loss near 10 (the maximum). Then I thought this code might just allow to calculate the gradient on dropout. I then found how to change the initial of pre-trained model, and I modify the AlexNet model without the first dropout function, but this time, I come across matrix size incorrect that at "output = models(images)" can't work. I tried various ways to modify the AlexNet model, but can't deal with this error.

As for GoogLeNet model, the result with fixed extractor is shown below: the best validation accuracy is 95.5%, with the loss of 0.1275, and train accuracy 95.25%, train loss 0.1293. The test accuracy is 85.5%. But the training time reduces to 1:43.

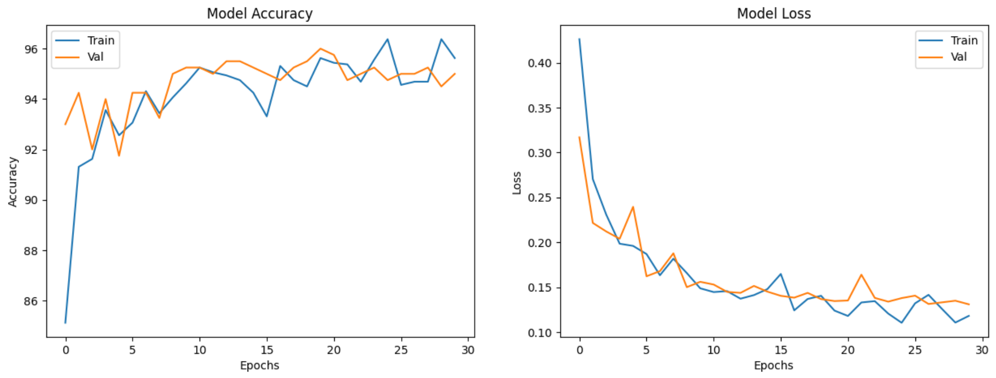

Fig.5. Result of GoogLeNet model with fixed extractor

4. *Task D: Comparison and Analysis*

In comparison with Task B and Task C, the major difference will be the steadily training process fine-tune model can achieve. And also with a large structure, the backward propagation will be much complex than fixed extractor, so the training time will be longer if we want to have a good training process.

As for testing, the fine-tune model's accuracy is worse than fixed extractor, I think if the testing dataset can be larger, the fine-tune model will be a better choice, because fine-tune can achieve higher-order feature representation in the base model in order to make them more relevant for the specific task.

5. Task E: Test Dataset Analysis

The first and the most important reason I assume, is that our dataset is too small, for only 1000 data, and cut it by 800/200, it's too small for a fine-tune model to have good test result. Although we can have good training results, with 800 training datasets, but with 200 testing datasets, it is unlikely to have good testing result. So, I modified the division to 700/300 and 600/400, after various training, I still can get good training results, but these times, I get better testing result with above 80% accuracy. But reduce the number of training dataset raises the problem of repeat training on some data, and this leads to incorrect training results. So, the correctness of testing result is highly suspicious.

And another assumption I come up is, the pre-trained model we use in the original code, the complexity is very high, that it uses a lot of time learning an image, so that it also learns about the noise in our dataset.

6. References

1. A.Krizhevsky, "One weird trick for parallelizing convolutional neural networks," (2014).
2. C.Szegedy, W.Liu, Y.Jia, P.Sermanet, S.Reed, D.Anguelov, D.Erhan, V.Vanhoucke, andA.Rabinovich, "Going deeper with convolutions," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. **07-12-June-2015**, 1–9 (2015).
3. I.Wahlang, P.Sharma, S.Sanyal, G.Saha, andA.Maji, "Deep learning techniques for classification of brain MRI," Int. J. Intell. Syst. Technol. Appl. **19**, 571 (2020).