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

馮渭中
Student ID: 107061222

1. Task A: Model Selection (20 pts)

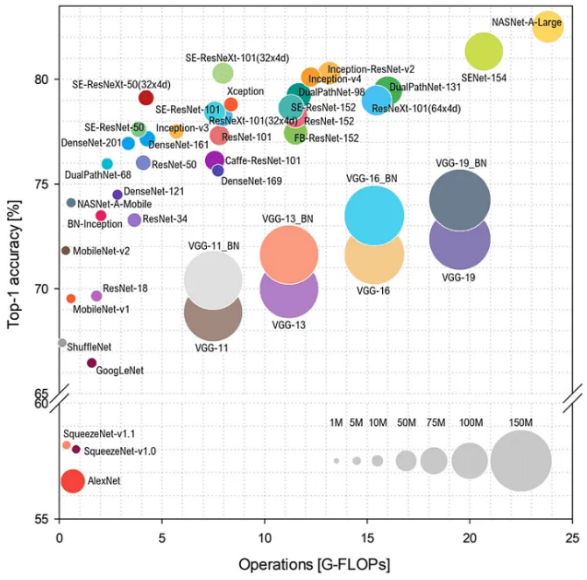
1.1 Model Choice

在HW4中, 我選擇DenseNet和ResNet18。

1.2 Explanation

這邊我用的pretrained model 為ResNet18和DenseNet, ResNet透過skip connection來解決訓練深層網路時會遇到梯度下降的問題, 綜合準確度(Top1 accuracy)與效率等考量, 試圖用ResNet50來進行chest X-ray dataset的分類。

DensNet則為ResNet的進階版, 將每一層都會跟其餘層做連接, 同時降低權種數。這些model都是用於訓練更深層的網路, 希望能在這次作業中看見他們的差異。



(Fig 00) Pretrained model-accuracy and operations

2. Task B: Fine-tuning the ConvNet (30 pts)

為了使輸入接近原model訓練條件, 這邊先調整data的channel number, std and mean。使的 Input data shape 為 (1000, 3, 224, 224)

```python
[11] import torch
     from torchvision import transforms
     from torch.utils.data import Dataset
     from PIL import Image

     class MedicalDataset(Dataset):
         def __init__(self, x, y, transform=None):
             self.x = x
             self.y = torch.from_numpy(y).long()

             # Define the default transformations if none are provided
             self.transform = transform or transforms.Compose([
                 transforms.Resize((224,224)),
                 transforms.Grayscale(num_output_channels=3),   # Converts 1-channel grayscale to 3-channel
                 transforms.ToTensor(),
                 transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
             ])

         def __len__(self):
             return len(self.x)

         def __getitem__(self, idx):
             # Convert the numpy array to a PIL image
             image = Image.fromarray(self.x[idx])

             # Apply the transformations to the PIL image
             if self.transform:
                 image = self.transform(image)

             return image, self.y[idx]
```

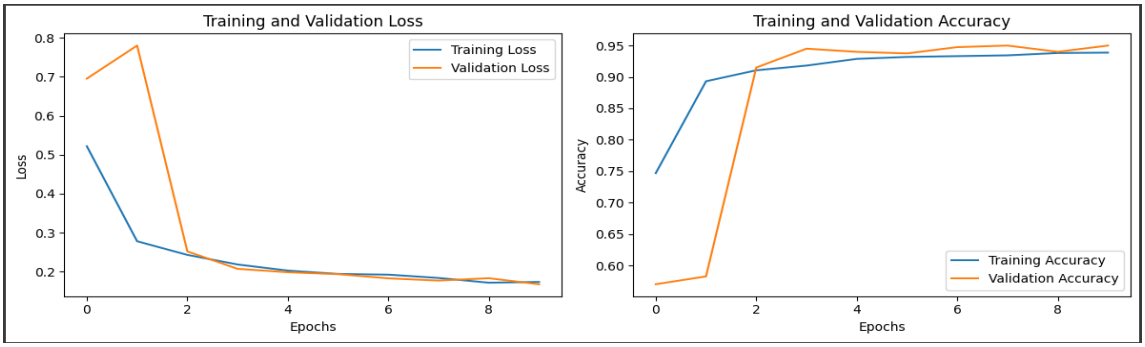
(Fig 01) Data Preprocessing for resNet18

Fine-tuning透過pretrain model加上新的dense layer, 透過training data重新更新模型權重, 這也代表著trainin dataset需要足夠的訓練資料, 才能在低learning rate的情況下避免 overfitting, 優點是對於與原pretrain model的訓練任務差異大時能更好的適應新任務。

2.1 Model Choice- ResNet18。

```python
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import models, transforms, datasets
from torch.utils.data import DataLoader

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

# 1. Load the pre-trained ResNet-18 model
model = models.resnet18(weights='ResNet18_Weights.IMAGENET1K_V1')

# 2. Modify the fully connected layer for binary classification
num_ftrs = model.fc.in_features
model.fc = nn.Linear(num_ftrs, 2)    # Change the output features to 2
model.to(device)

# 3. Define a loss function and optimizer
criterion = nn.CrossEntropyLoss()

# Only parameters of the final layer are being optimized as
optimizer = optim.SGD(model.fc.parameters(), lr=0.001, momentum=0.9)
```

(Fig 02) ResNet18 + binary classification

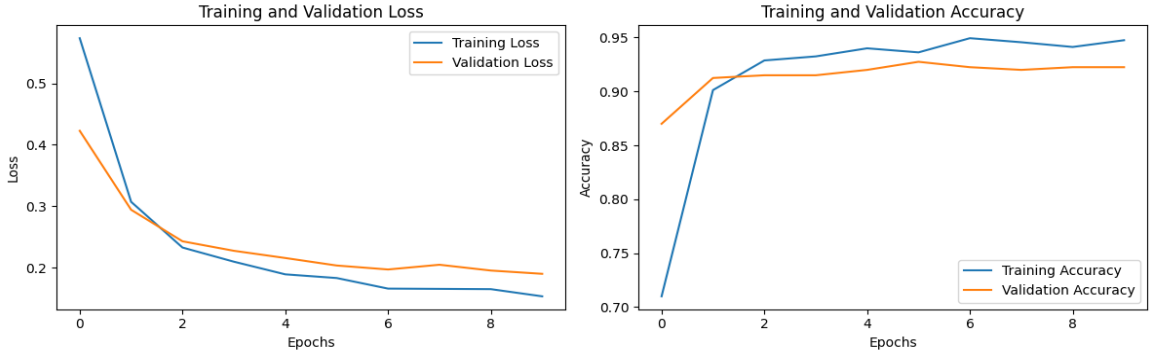


(Fig 03) *ResNet18-fine tune_*Loss and Accuracy

這邊可以看到在10 epochs內loss和accuracy優化的速度極快, 約3 epochs就已經達到91%的準確度。

2.2 Model Choice- DenseNet

DenseNet這邊可以看到loss的下降更為穩定, 然而相較於ResNet18有些underfitting的情快, 但可以看到兩種pretrian model透過fine tune的方法都可以明顯優於HW2的訓練結果。



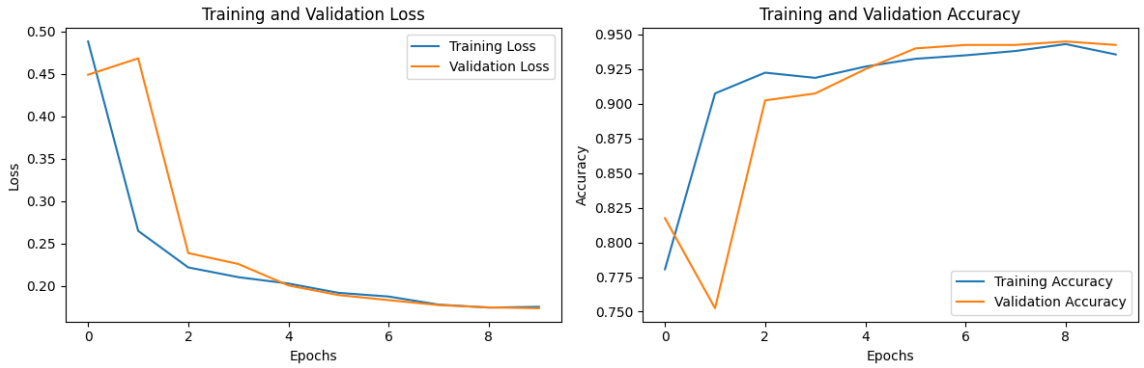
(Fig 04) *DenseNet-fine tune_*Loss and Accuracy

3 Task C: ConvNet as Fixed Feature Extractor (30 pts)

Feature_Extraction這個方法在訓練過程最明顯的差異在於"訓練速度快", 由於不更新pretrained layer的權重, 而只訓練用於分類的dense layer, 在運算速度能力有限的時候可以有效的訓練模型。

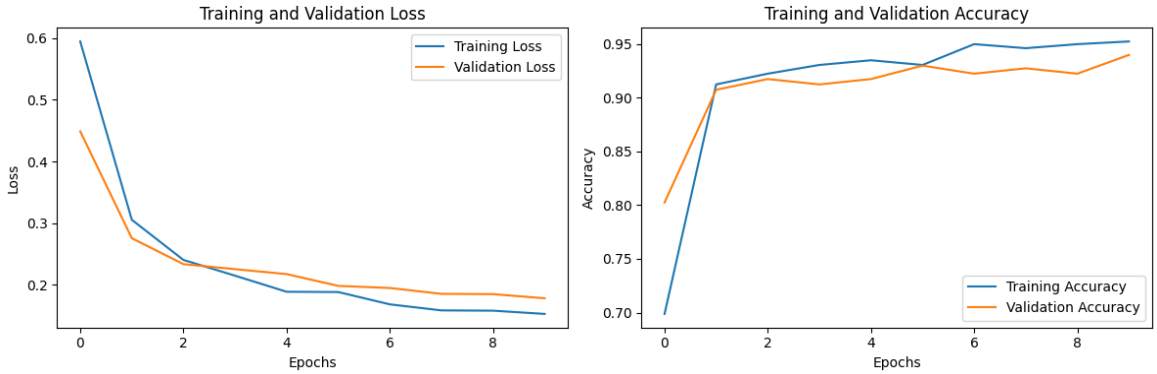
而另一個好處則是當訓練資料過少時, 當pretrained model可以有效捕捉圖片特徵時, 使用Feature Extraction可以有效避免overfitting。

3.1 Discussion - ResNet18



(Fig 05) *ResNet18-feature extraction_*Loss and Accuracy

3.2 Discussion - DenseNet



(Fig 06) *DenseNet-feature extraction_*Loss and Accuracy

4 Task D: Comparison and Analysis (10 pts)

4.1 Discussion

就訓練結果而言，兩種方法best accuracy都接近95%且loss也都下降到0.2左右，就表現上十分接近。然而最明顯的是訓練時間，使用fine tuning的方法明顯較慢，由於更新的權重較多對運算時間較多且效率較差。以下針對兩種方法進行比較:

(1) Flexibility to new data : Fine tuning能更好的適應差異大的訓練任務，而Featrue exractor要求任務性質接近。
(2) Training time and compute : Fine tuning 訓練速度較耗時，這也與我訓練觀察到的結果相同。
(3) Risk of overfitting : 在資料數量較少時，Fine tuning overfitting的風險較高，然而若資料數量夠且訓練的好，Fine tuning的表現會優於 Feature extractor。

5 Task E: Test Dataset Analysis (10 pts)

5.1 Discussion

總共2000筆資料有些過少，若透過更深層複雜的網路層訓練會碰到overfitting的問題，若能透過適當的data augmentation可以得到較好的訓練效果。