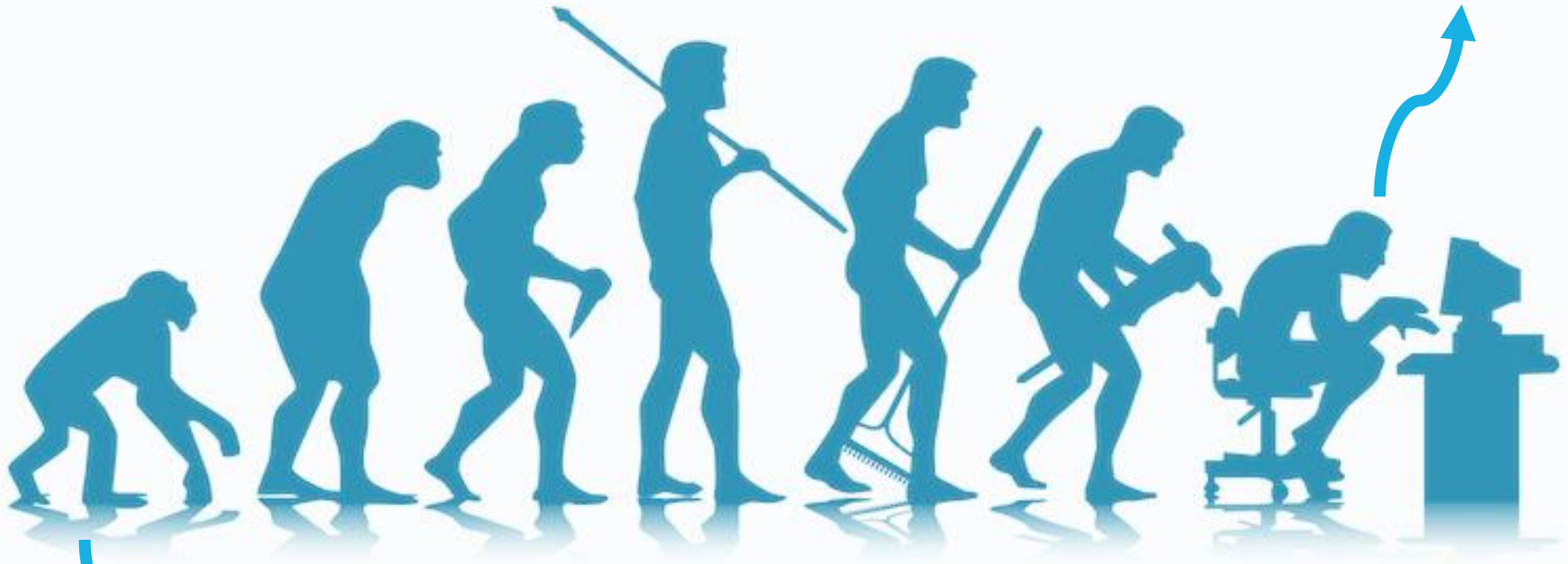
人工智慧實務期末報告

-baseline

baseline-組員及分工

~~來上爽課(打程式打得很爽)~~、程式設計、優化:**陳彥瑋**



王柏文:簡報設計、上台報告、概念構想、資料搜尋、協助程式設計

HW3 CNN圖片判別

AIoT Final Presentation Leader Board 2

Order by Accuracy then by submitted time.

#	Group Name	Student1	Student2	Student3	Submit times	Submit time	Accuracy
1	狗勾都比你會打程式	簡晟恩	鄭善溯	宋政崴	23	2022/01/01 21:50:51	0.81356
---	Strong Baseline						0.81011
2	Baseline	陳彥瑾	王柏文		13	2022/01/04 12:17:20	0.71646
3	我今天一定要爽死	鄭永其	蔡文成		34	2021/12/31 21:35:28	0.62295

Homework 3 - Convolutional Neural Network(CNN, 卷積神經網路)

目標

- 學會如使用 CNN 做影像分類
- 使用影像增強改進預測準確度
- 學會如何使用 Unlabeled Data(未分類資料) 和 Semi-Supervised 的收益

Baselines

- Easy
  - 利用範例程式碼建立一個 CNN 模型，並使用已標註的圖片
- Medium
  - 修改模型參數、使用不同模型或使用影像增強，再利用已標註的圖片改進準確率
  - 你可以透過增加一些程式碼到範例程式碼中達到此目標
- Strong
  - 使用未標註的資料來改進準確率
  - 這裡可以使用未標註的 testing data
  - 提示: 使用 semi-supervised learning, self-supervised learning

資料處理方式-影像增強

隨機鏡面翻轉(對Y軸)

隨機位移
(角度、位移量、縮放比例、錯切)

影像調整(亮度、對比、飽和)

```
train_tfm = transforms.Compose([
        # Resize the image into a fixed shape (height = width = 128)
        transforms.Resize((128, 128)),
        # You may add some transforms here.
        transforms.RandomHorizontalFlip(p=0.5),
        transforms.RandomAffine(30, translate=(0.3, 0.3), scale=(0.8, 1.2), shear=0.3),
        transforms.ColorJitter(brightness=(0.5, 1.5), contrast=(0.5, 1.5), saturation=(0.5, 1.5)),
        transforms.RandomRotation(30),
        # ToTensor() should be the last one of the transforms.
        transforms.ToTensor(),
])
```

效果:使訓練資料的圖片多樣性增加、不易Over firtting

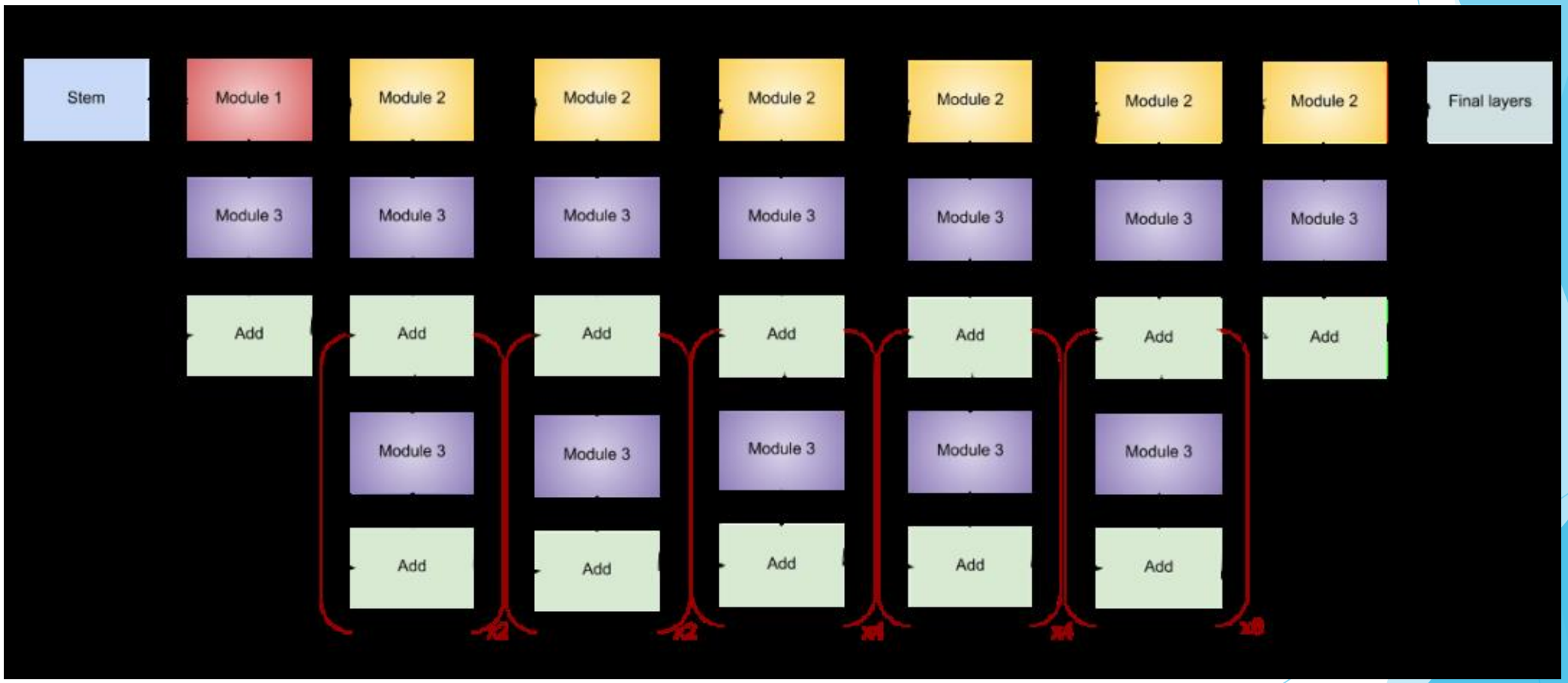
NN架構介紹-efficientNet_b4

```
12 #  MY  MODIFY
13 model  =  torchvision.models.efficientnet_b4(pretrained=False).to(device)
14 model.classifier  =  nn.Sequential(
15         nn.Dropout(p=0.4,  inplace=True),
16         nn.Linear(in_features=1792,  out_features=1000),
17         nn.Dropout(p=0.4,  inplace=True),
18         nn.Linear(in_features=1000,  out_features=11)
19     ).to(device)
20 model.device  =  device
```

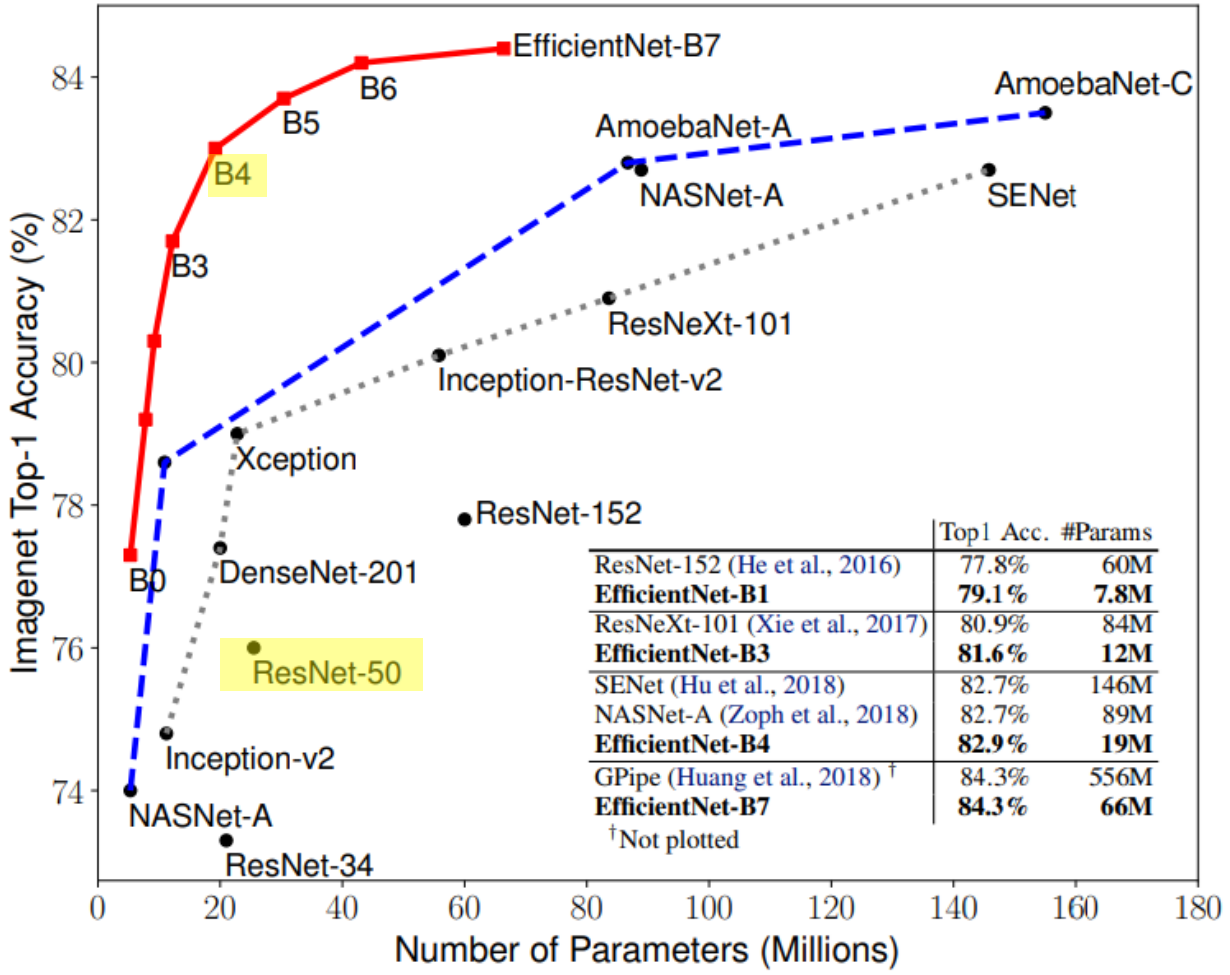
比較資料集:food101

Rank	Model	Accuracy	FLOPS	PARAMS	Top 1 Accuracy	Extra Training Data	Paper	Code	Result	Year	Tags
1	**EffNet-L2** (SAM)	96.18				✓	Sharpness-Aware Minimization for Efficiently Improving Generalization			2020	
2	**ALIGN**	95.88				✓	Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision			2021	
3	**Grafit** (RegNet-8GF)	93.7				×	Grafit: Learning fine-grained image representations with coarse labels			2020	
4	**EfficientNet-B7**	93.0				×	EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks			2019	EfficientNet
5	**Assemble-ResNet-FGVC-50**	92.5				×	Compounding the Performance Improvements of Assembled Techniques in a Convolutional Neural Network			2020	

Architecture of efficientNet_b4



Why EfficientNet ?



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

效果：同樣Parameters下，表現遠高於ResNet

半監督式學習

```
def get_pseudo_labels(dataset, model, threshold=0.87):
    pseudo_label_folder = "./pseudo_label/"

        # ---------- TODO ----------
        # Filter the data and construct a new dataset.
        for i in range(probs.shape[0]):
          if(torch.max(probs[i]) >= threshold):
            pseudo_label = probs[i].argmax(dim=-1)
            im = transforms.ToPILImage()(img[i])
            if(os.path.isdir(pseudo_label_folder+f"{pseudo_label.item():02d}/") == False):
              os.mkdir(pseudo_label_folder+f"{pseudo_label.item():02d}/")
            im.save(pseudo_label_folder+f"{pseudo_label.item():02d}/{ct}_{i}.jpg")

        pbar.update(1)
        ct += 1

    # FileNotFoundError
    try:
      dataset = DatasetFolder(pseudo_label_folder, loader=lambda x: Image.open(x), extensions="jpg", transform=train_tfm)
    except FileNotFoundError:
      dataset = None

    # # Turn off the eval mode.
    model.train()
    return dataset
```

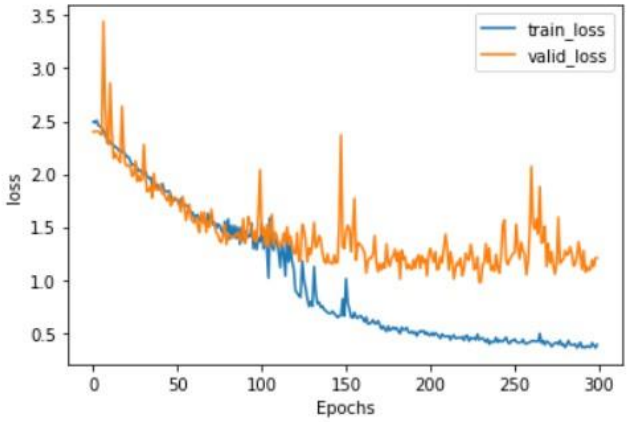
```
# ---------- TODO ----------
# In each epoch, relabel the unlabeled dataset for semi-supervised learning.
# Then you can combine the labeled dataset and pseudo-labeled dataset for the training.
semi_count = 0
if do_semi:
  # Obtain pseudo-labels for unlabeled data using trained model.
  pseudo_set = get_pseudo_labels(unlabeled_set, model)
  if(pseudo_set != None):
    # Construct a new dataset and a data loader for training.
    # This is used in semi-supervised learning only.
    concat_dataset = ConcatDataset([train_set, pseudo_set])
    train_loader = DataLoader(concat_dataset, batch_size=batch_size, shuffle=True, num_workers=1, pin_memory=True)
    semi_count = len(pseudo_set)
    print(f"Epoch {epoch} has {semi_count} data to do semi-supervise")
  else:
    train_loader = DataLoader(train_set, batch_size=batch_size, shuffle=True, num_workers=1, pin_memory=True)
    print(f"Epoch {epoch} has no data doing semi-supervise")
```

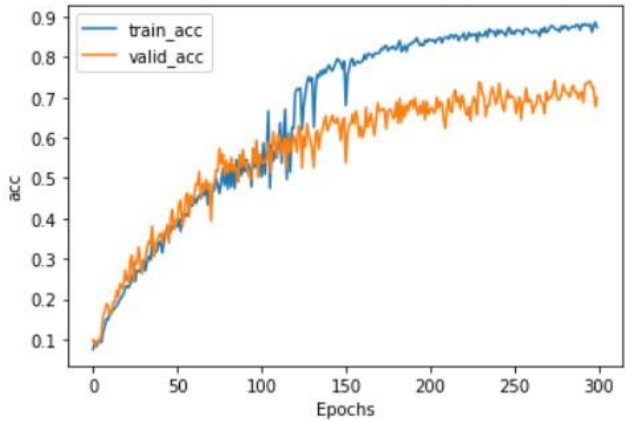
訓練方式

```
optimizer = torch.optim.Adam(model.parameters(), lr=0.0003, weight_decay=1e-5)
```

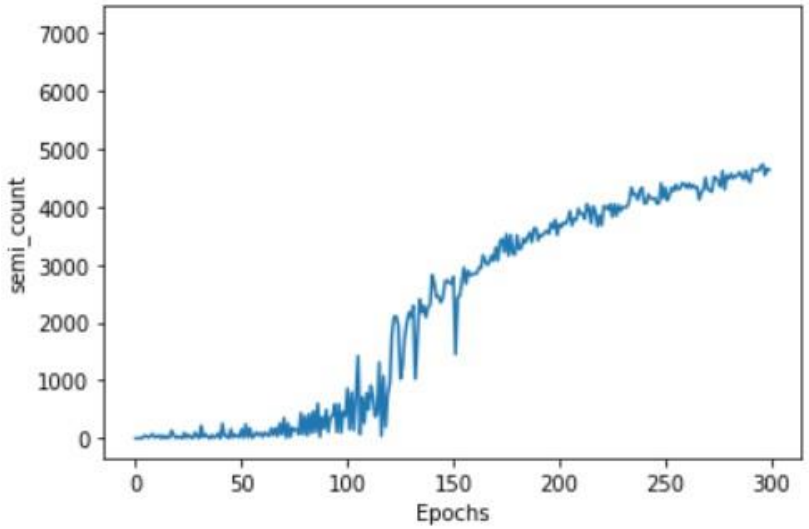
```
# The number of training epochs.
n_epochs = 300 # 80
```

訓練時間: 13hrs & 25 mins
每個epoch平均訓練時間: 2mins & 40s



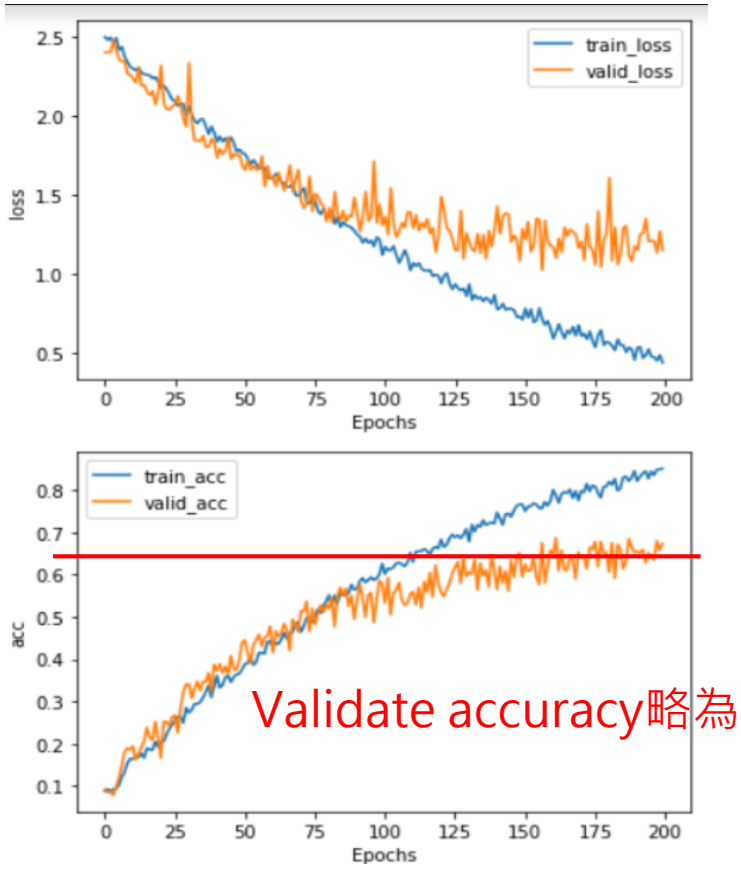


發現 : pseudo label之數量和training accuracy之高低隨epochs增加的曲線，兩者非常相似

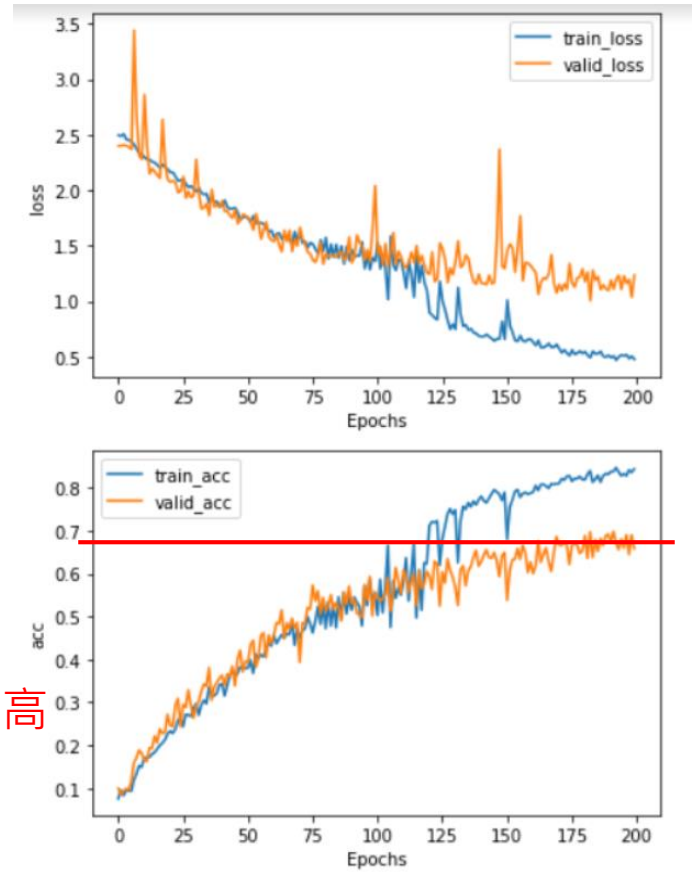


半督促式學習前後比較

前:



後:



Validate accuracy略為提高

自動進行資料備份

```python
class History():
  def __init__(self):
    self.done_epoch = -1 # len(list) - 1
    self.metrics = {"train_loss":[], "train_acc":[], "valid_loss":[], "valid_acc":[], "semi_count":[]}
  def store_record(self, train_loss, train_acc, valid_loss, valid_acc, semi_count):
    self.metrics["train_loss"].append(train_loss)
    self.metrics["train_acc"].append(train_acc)
    self.metrics["valid_loss"].append(valid_loss)
    self.metrics["valid_acc"].append(valid_acc)
    self.metrics["semi_count"].append(semi_count)
    self.done_epoch += 1
  def load(self, path):
    if(os.path.isfile(path)):
      with open(path, 'rb') as f:
        self.metrics = pickle.load(f)
      self.done_epoch = len(self.metrics["train_loss"]) - 1
      return 1
    else:
      return -1
  def store(self, path):
    with open(path, 'wb') as f:
      pickle.dump(self.metrics, f)
```

```python
store_path = "/content/drive/Shareddrives/未命名的共用雲端硬碟/AICourse_hw/hw3_semi_full/"
# store_path = "./temp/"
if(store_path[-1] != "/"):
  store_path += "/"
if(os.path.isdir(store_path) == False):
  os.mkdir(store_path)
```

```python
history = History()
load_fg = -1
if(os.path.isfile(store_path+"history.pkl")):
  load_fg = history.load(store_path+"history.pkl")
  checkpoint = torch.load(store_path+f"model_{history.done_epoch}.pt")
  model.load_state_dict(checkpoint['model_state_dict'])
  optimizer.load_state_dict(checkpoint['optimizer_state_dict'])

if(load_fg == 1):
  print(f"Load success! Training after {history.done_epoch+1}")

for epoch in range(n_epochs):
  if(load_fg==1 and epoch <= history.done_epoch):
    semi_count = history.metrics['semi_count'][epoch]
    if(semi_count == 0):
      print(f"Epoch {epoch} has no data doing semi-supervise")
    else:
      print(f"Epoch {epoch} has {semi_count} data to do semi-supervise")
```

Thanks for listening!