Deep Learning for Computer Vision

NTU, Fall 2023, homework1

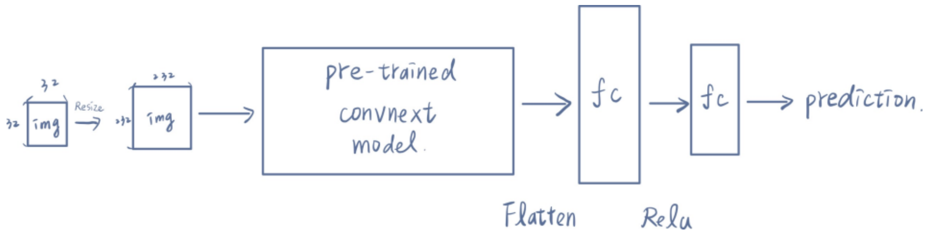
電機所碩一 謝宗翰 R12921A10

● **Problem 1: Image Classification** (25%)

1. *Draw the network architecture of method A or B*

```python
model = convnext_base(weights=ConvNeXt_Base_Weights.DEFAULT)

model.classifier = nn.Sequential(
    nn.Flatten(), nn.Linear(1024, 500), nn.ReLU(), nn.Linear(500, 50)
)
```



2. *Report accuracy of your models (both A, B) on the validation set.*

Accuracy	
A	B
30.7%	90.2%

3. *Report your implementation details of model A.*

```python
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()  # 可調用nn.Moudule的函數

        # 第一個卷積層
        self.conv1 = nn.Conv2d(in_channels=3, out_channels=32, kernel_size=3, padding=1)
        self.relu1 = nn.ReLU()
        self.pool1 = nn.MaxPool2d(kernel_size=2, stride=2)

        # 第二個卷積層
        self.conv2 = nn.Conv2d(
            in_channels=32, out_channels=64, kernel_size=3, padding=1
        )
        self.relu2 = nn.ReLU()
        self.pool2 = nn.MaxPool2d(kernel_size=2, stride=2)

        # 全連接層
        self.fc1 = nn.Linear(in_features=29696, out_features=128)
        self.relu3 = nn.ReLU()
        self.fc2 = nn.Linear(in_features=128, out_features=50)  # 50類

        self.drop25 = nn.Dropout2d(0.25)
```

```python
    def forward(self, x):
        x = self.conv1(x)
        x = self.relu1(x)
        x = self.pool1(x)

        x = self.conv2(x)
        x = self.relu2(x)
        x = self.pool2(x)

        x = x.view(x.size(0), -1)  # 展平特徵圖
        x = self.fc1(x)
        return x
```

A Model				
Learning Rate	Batch Size	Optimizer	Epochs	Loss Func
0.0003	32	Adam	80	Cross Entropy
Train Data augmentation		Valid Data augmentation		
No		No		

B Model				
Learning Rate	Batch Size	Optimizer	Epochs	Loss Func
0.0005	128	Adam	50	Cross Entropy

Train Data augmentation

```python
basic_transform = [
    # additional data argument
    trns.RandomResizedCrop(224, scale=(0.08, 1.0), ratio=(3.0 / 4.0, 4.0 / 3.0)),
    trns.RandomHorizontalFlip(),  # 水平翻轉
    trns.ColorJitter(brightness=0.5, contrast=0.5, hue=0.5),
]

transform_train = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.RandomChoice(basic_transform),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

Valid Data augmentation

```python
transform_val = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

4. *Report your alternative model or method in B, and describe its difference from model A.*

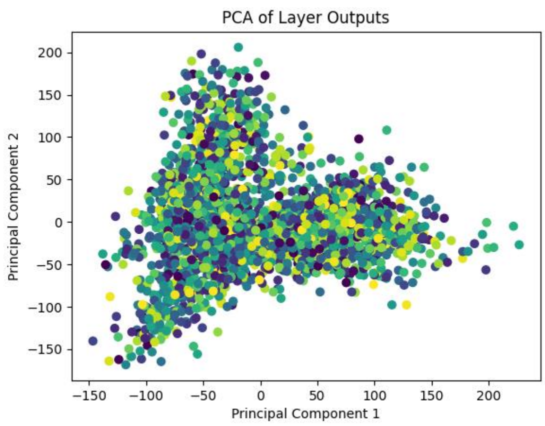
我的 B 用 torchvision model 的 ConvNext 進行 finetuning，把第 7 層前的

weight 都 freeze 住，只 train 倒數幾層捲積層及 Classifier。

```python
for i, (name, param) in enumerate(model.named_parameters()):
    param.requires_grad = False
    if "classifier" in name or "7" in name:
        param.requires_grad = True
```

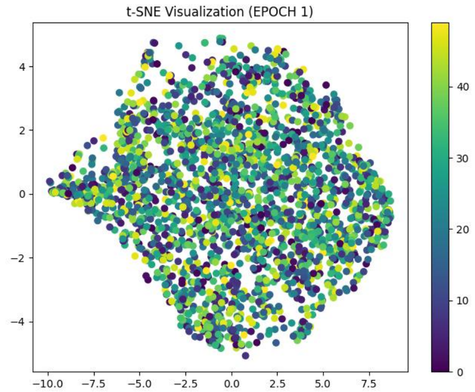
跟 A 比較不一樣的地方是，B 是用 pre-train 的 weight，著重在 Classifier 的訓練，讓 pre-train model 能對提供的 dataset 進行分類。B 模型比我自己手刻的 Net 深很多，效果比我的 A 好非常多。

5. *Visualize the learned visual representations of model A on the validation set by implementing PCA (Principal Component Analysis) on the output of the second last layer. Briefly explain your result of the PCA visualization.*

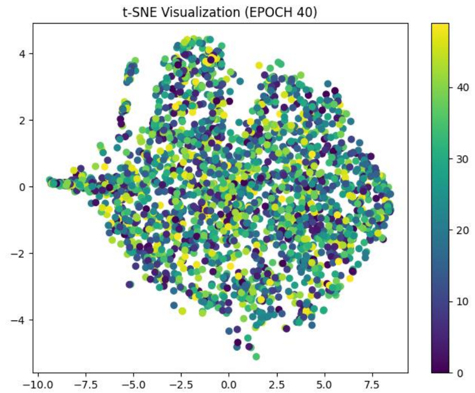
Model A PCA Visualization

可以看出可以略為的去劃出一條分割線，代表訓練的結果還行。但是以各個顏色(label)的分布來看，並沒有分得很開，大多數的 label 都是交疊在一起的。

6. *Visualize the learned visual representation of model A, again on the output of the second last layer, but using t-SNE (t-distributed Stochastic Neighbor Embedding) instead. Depict your visualization from three different epochs including the first one and the last one. Briefly explain the above results.*

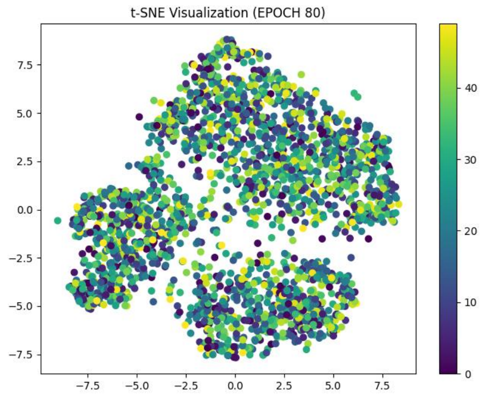
Model A t-SNE Visualization
Epoch = 1



Epoch = 40



Epoch = 80

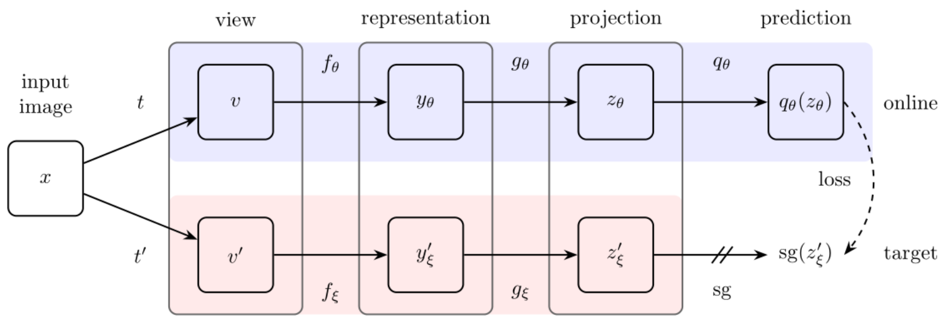


隨著 Epoch 越來越大，可以看出各點開始由中心開始往外擴，由訓練到 Epoch = 80 時非常明顯。但顏色跟顏色之間也都是交疊在一起，無明顯區別。

● Problem 2: Self-Supervised Pre-training for Image Classification

1. *Describe the implementation details of your SSL method for pre-training the ResNet50 backbone.*

我的 SSL 採用助教提供的方法(BYOL)



上圖為 BYOL 的架構，他有兩個網路，分別為 Target 及 Online

(1) 先對 Input x 做 data argmentation，得到 t 及 t'

(2) 將 t 輸入至 Online 網路中經f_θ 提取特徵，得到特徵向量y_θ。同時將t'輸入到 target 中，經f_ε提取特徵，得到特徵向量y'_ε。

(3) y_θ 經過 MLP 網路g_θ，得到z_θ。而y'_ε 經過 MLP 網路g_ε，得到z'_ε。

(4) z_θ 經過 MLP 網路q_θ ，得到$q_\theta(z_\theta)$，與z'_ε計算 loss

(5) $loss$ 參照 BYOL 的論文

$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}(z_\theta) - \overline{z}'_\xi\right\|_2^2 = 2 - 2\cdot \frac{\langle q_\theta(z_\theta), z'_\xi\rangle}{\left\|q_\theta(z_\theta)\right\|_2 \cdot \left\|z'_\xi\right\|_2}.$$

SSL Model (BYOL)				
Learning Rate	Batch Size	Optimizer	Epochs	Loss Func
0.01	256	Adam	10	Focal Loss
Train Data augmentation				

```
basic_transform = [
    # additional data argument
    trns.RandomResizedCrop(224, scale=(0.08, 1.0), ratio=(3.0 / 4.0, 4.0 / 3.0)),
    trns.RandomHorizontalFlip(),  # 水平翻轉
    trns.ColorJitter(brightness=0.5, contrast=0.5, hue=0.5),
]

transform_train = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.RandomChoice(basic_transform),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

Valid Data augmentation

```python
transform_val = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

2. *Please conduct the Image classification on Office-Home dataset as the downstream task. Also, please complete the following Table, which contains different image classification setting, and discuss/analyze the results.*

SSL Info				
Learning Rate	Batch Size	Optimizer	Epochs	Backbone
0.0005	128	Adam	10	Resnet50
Train Data augmentation				

```python
basic_trans = trns.Compose(
    [
        trns.Resize([128, 128]),
        trns.RandomHorizontalFlip(),
        trns.TrivialAugmentWide(),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

Fine Tuning Info (ABCDE 都用此規格)				
Learning Rate	Batch Size	Optimizer	Epochs	Loss Func
0.0005	64	Adam	100	Cross Entropy
Train Data augmentation				

```python
basic_transform = [
    # additional data argument
    trns.RandomResizedCrop(224, scale=(0.08, 1.0), ratio=(3.0 / 4.0, 4.0 / 3.0)),
    trns.RandomHorizontalFlip(),  # 水平翻轉
    trns.ColorJitter(brightness=0.5, contrast=0.5, hue=0.5),
]

transform_train = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.RandomChoice(basic_transform),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

Valid Data augmentation

```python
transform_val = trns.Compose(
    [
        trns.Resize((232, 232), interpolation=trns.InterpolationMode.BICUBIC),
        trns.CenterCrop((224, 224)),
        trns.ToTensor(),
        trns.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
    ]
)
```

Discuss/Analyze the results

A	B	C	D	E
41%	59.5%	49.6%	53.9%	34%

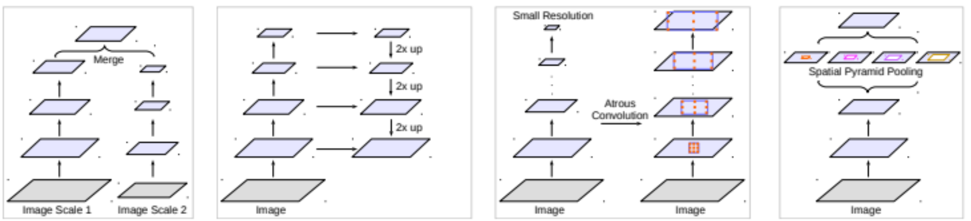
- A 為單純用 ResNet50 訓練整個 model 的結果，結果比 E 只訓練 Classifier 的好。
- B 是用助教提供的 pre-train weight 去訓練整個 ResNet50，結果優於我寫的 SSL (BYOL)，也是所有選項中最高的。
- 我用 BYOL 弄出的 C 的結果比助教提供的 pre-train weight 的 B 與 D 都低，但都比沒用 pre-train weight 訓練的模型高。
- D 與 E 皆比 B 與 C 還低，代表單純只訓練 classifier 是完全不夠的，觀察 E 與 B 的差距可見，我的 SSL(BYOL)提供的效果有限。

● Problem 3: Semantic Segmentation Task Definition

1. *Draw the network architecture of your VGG16-FCN32s model (model A).*

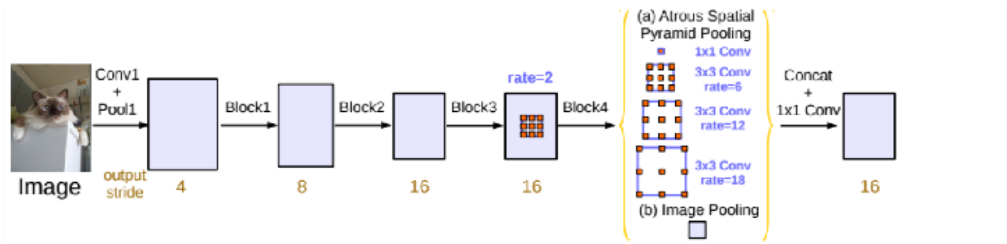
2. *Draw the network architecture of the improved model (model B) and explain it differs from your VGG16-FCN32s model.*

B model 我用 Troch Vision 的 DeepLab3_Resnet50 來 train



(a) Image Pyramid (b) Encoder-Decoder (c) Deeper w. Atrous Convolution (d) Spatial Pyramid Pooling

Figure 2. Alternative architectures to capture multi-scale context.



(原 paper 的架構)

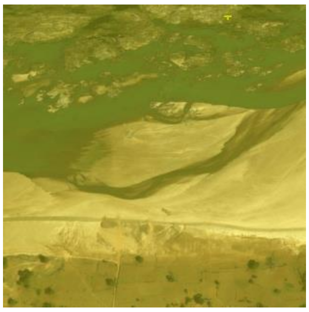
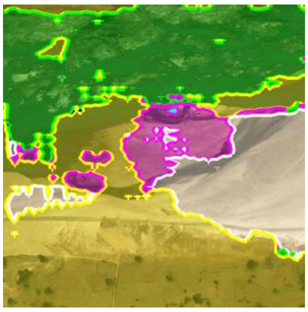
B 跟 A 的架構大體上是差不多的，一個 Encoder 與一個 Decoder，差別不同在於他們處理 Layer 的方式不一樣。A 的 Encoder 是用 Vgg16，B 則是用 Resnet50。

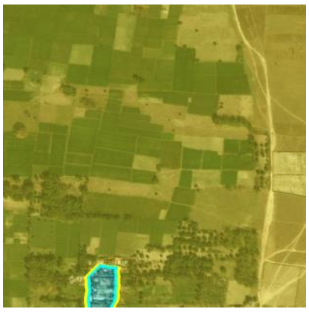
DeepLabv3 使用了空洞捲積（Dilated Convolution）作為其主要特點，這有助於捕獲多尺度的語義資訊。它還包括了空間金字塔池化（ASPP）模組，用於在不同尺度上捕獲上下文資訊。DeepLabv3 基於其先進的架構和多尺度上下文資訊捕獲，通常能夠獲得更好的語義分割性能。 它在許多分割任務中表現出色，特別是在複雜場景和小目標分割方面。

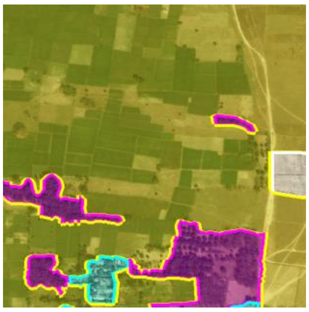
3. *Report mIoUs of two models on the validation set.*

VGG16-FCN32 mIoU	DeepLabv3_Resnet50 mIoU
56.4%	72.9%

4. *Show the predicted segmentation mask of "validation/0013_sat.jpg" , "validation/0062_sat.jpg", "validation/0104_sat.jpg" during the early, middle, and the final stage during the training process of the improved model.*

validation/0013_sat.jpg	
Epoch = 1	Epoch = 50
	
Epoch = 100	Ans
	

validation/0062_sat.jpg	
Epoch = 1	Epoch = 50
	
Epoch = 100	Ans

<table>
<tr><th colspan="2">validation/0104_sat.jpg</th></tr>
<tr><th>Epoch = 1</th><th>Epoch = 50</th></tr>
<tr><td></td><td></td></tr>
<tr><th>Epoch = 100</th><th>Ans</th></tr>
<tr><td></td><td></td></tr>
</table>