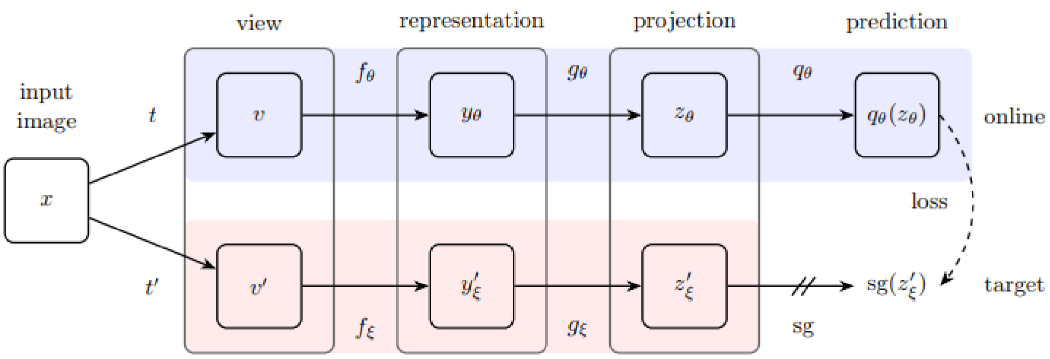
R12521504 吳玉文

Problem 1

1. *Describe the implementation details of your SSL method for pre-training the ResNet50 backbone. (including but not limited to the name of the SSL method & data augmentation techniques you used, learning rate schedule, optimizer, and batch size setting for this pre-training phase)*

採用助教所提供的 BYOL 方法，架構如下：



對輸入圖像 x 進行 data augmentation，得到 t 以及 t'

> 將 t 輸入到 online 網路經過 f_θ 進行特徵提取得到 y_θ ; t'輸入到 target 網路經過 f_ξ 進行特徵提取得到 y'_ξ

> 將 y_θ 經過 MLP 網路 g_θ 處理得到 z_θ ; y'_ξ 經過 MLP 網路 g_ξ 處理得到 z'_ξ

> z_θ 再經過 MLP 網路 q_θ 得到 $q_\theta(z_\theta)$，並與 z'_ξ 進行比較以計算 loss

> Loss function:

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

SSL 訓練細節描述：

> Method: BYOL

> Batch Size: 32

> Optimizer: Adam

> Learning Rate: 1e-2

> Learning Rates Scheduler: CosineAnnealingWarmRestarts

> Num of Epochs: 50+50 (train 完 50 個 epochs 後的權重再 train 50 次)

> Data Augmentation:

```
transform_train = transforms.Compose([
    transforms.Resize(128, interpolation=transforms.InterpolationMode.BICUBIC),
    transforms.CenterCrop(128),
    transforms.TrivialAugmentWide(),
    transforms.ToTensor(),
    transforms.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.*

訓練細節描述：

> Batch Size: 32

> Optimizer: Adam

> Learning Rate: 1e-3

> Learning Rates Scheduler: CosineAnnealingWarmRestarts

> Num of Epochs: 100

> Data Augmentation:

```
transform_train = transforms.Compose([
    transforms.Resize(128, interpolation=transforms.InterpolationMode.BICUBIC),
    transforms.CenterCrop(128),
    transforms.TrivialAugmentWide(),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

> Results:

Setting	Validation Accuracy
A	52.22%
B	53.20%
C	42.86%
D	34.73%
E	28.57%

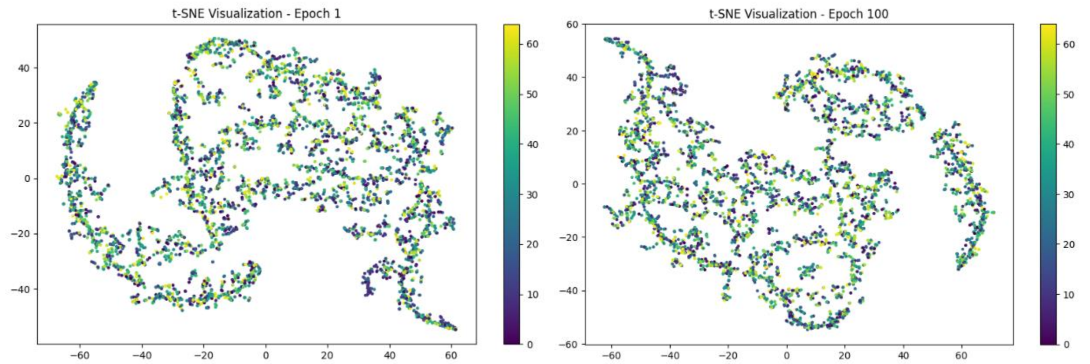
> Discussion:

有載入助教 pre-train weight 的模型(Setting B)結果比起沒有 pre-train weight 的(Setting A)更為準確，而且不用訓練到太多的 epochs 其實就可以

得到良好的結果了，顯示出在有良好的 pre-train weight 下，模型會有比較好的結果。但我自己訓練出的 pre-train weight(Setting C)，在 fine tune 的部分表現並沒有比較好，有可能是因為 pre-train 不夠久，導致 loss 還沒有完全收斂的緣故。但因為比起 Setting A，training accuracy 跟 validation accuracy 在 Setting C 的 training 過程中差距並沒有那麼大，所以雖然 validation accuracy 並沒有上升，但 pre-train weight 有讓整個模型較為 robust，防止在 training set 上 overfitting。而 Setting D/E 與 Setting B/C 進行比較的結果可以看出，固定 backbone 會使得模型沒辦法完整接收不一樣 training data 的資訊，使得準確率降低。

3. *Visualize the learned visual representation of setting C on the train set by implementing t-SNE (t-distributed Stochastic Neighbor Embedding) on the output of the second last layer. Depict your visualization from both the first and the last epochs. Briefly explain the results.*

> First and Last Epoch:

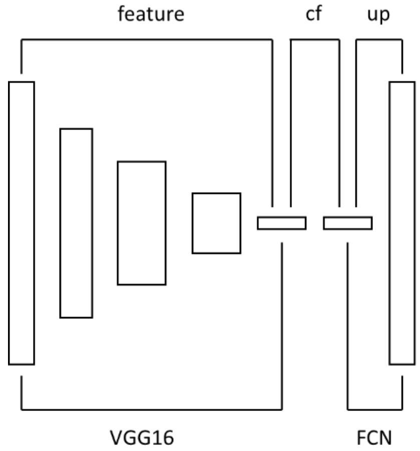


> Discussion:

雖然不同顏色的點還沒有辨識的很明顯，但不同類別的距離可以明顯看出有被拉開，顯示出模型能夠透過訓練逐漸理解不同 label 的圖片之間的差異。

Problem 2

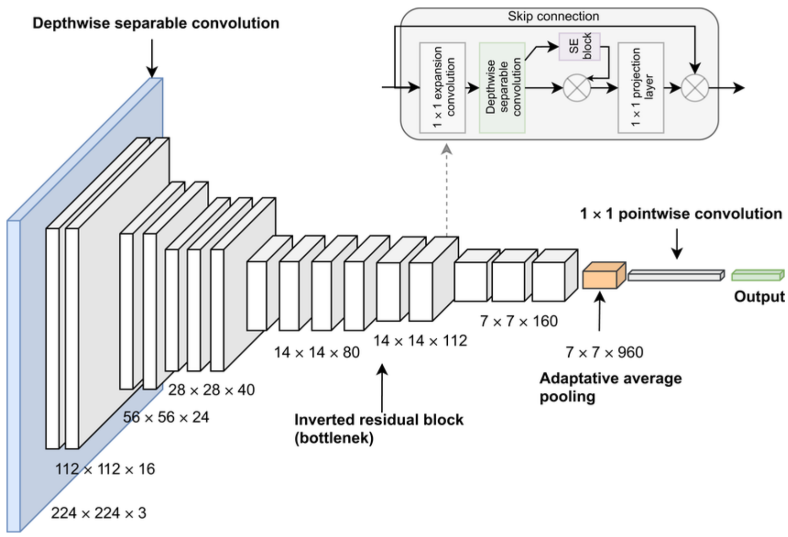
1. *Draw the network architecture of your VGG16-FCN32s model (model A).*




```python
def __init__(self, num_classes):
    super(VGG16_FCN32s, self).__init__()

    vgg = models.vgg16(weights='DEFAULT')

    self.features = vgg.features

    self.classifier = nn.Sequential(
        nn.Conv2d(512, 1024, kernel_size=7),
        nn.ReLU(inplace=True),
        nn.Dropout2d(),
        nn.Conv2d(1024, num_classes, kernel_size=1),
        nn.ReLU(inplace=True)
    )

    self.upscore = nn.ConvTranspose2d(
        num_classes,
        num_classes,
        kernel_size=44,
        stride=52,
        padding=0,
        bias=False
    )
```

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

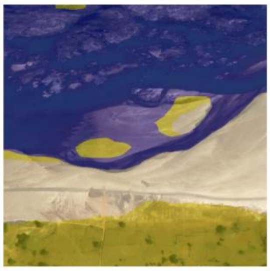
我使用 segmentation_models_pytorch 的 timm-mobilenetv3_large_100 進行 model B 的訓練。與 model A 相比也同樣是 Encoder（如下圖所示）加上 Decoder 的架構，但 mobilenetv3 在處理 Layer 時的做法更加進步，使用了 Inverted Residual、Squeeze-and-Excitation 模塊，並結合了 Hardswish 激活函數。這些改進讓 model B 在保持高效的同時，能夠更好地捕捉和表達複雜的特徵。此外，model B 還透過更輕量化的卷積操作和優化的網路設計提高了性能，使其能在計算資源受限的設備上表現尤佳。

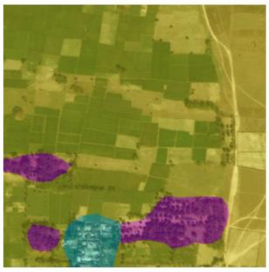
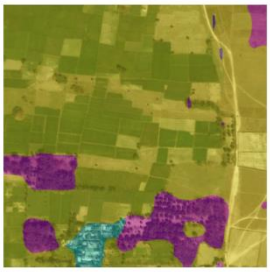
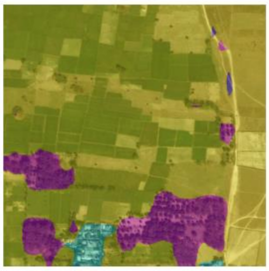
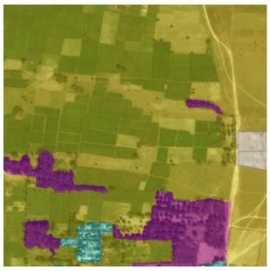


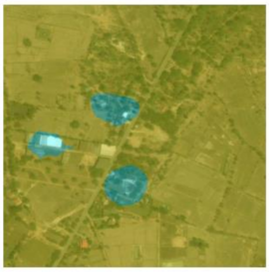
3. *Report mIoUs of two models on the validation set.*

Model A	Model B
39.5%	72.6%

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.*

<table>
<tr><td colspan="2">validation/0013_sat.jpg</td></tr>
<tr><td>Epoch 1</td><td>Epoch 75</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Epoch 150</td><td>Label</td></tr>
<tr><td></td><td></td></tr>
</table>

<table>
<tr><td colspan="2">validation/0062_sat.jpg</td></tr>
<tr><td>Epoch 1</td><td>Epoch 75</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Epoch 150</td><td>Label</td></tr>
<tr><td></td><td></td></tr>
</table>

<table>
<tr><td colspan="2">validation/0104_sat.jpg</td></tr>
<tr><td>Epoch 1</td><td>Epoch 75</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Epoch 150</td><td>Label</td></tr>
<tr><td></td><td></td></tr>
</table>

5. *Use segment anything model (SAM) to segment three of the images in the validation dataset, report the result images and the method you use.*

我選擇 0050_sat.jpg、0130_sat.jpg、0256_sat.jpg 進行 segment，model 使用 vit_h，呈現出的結果如下：

0050_sat.jpg

0130_sat.jpg

0256_sat.jpg