

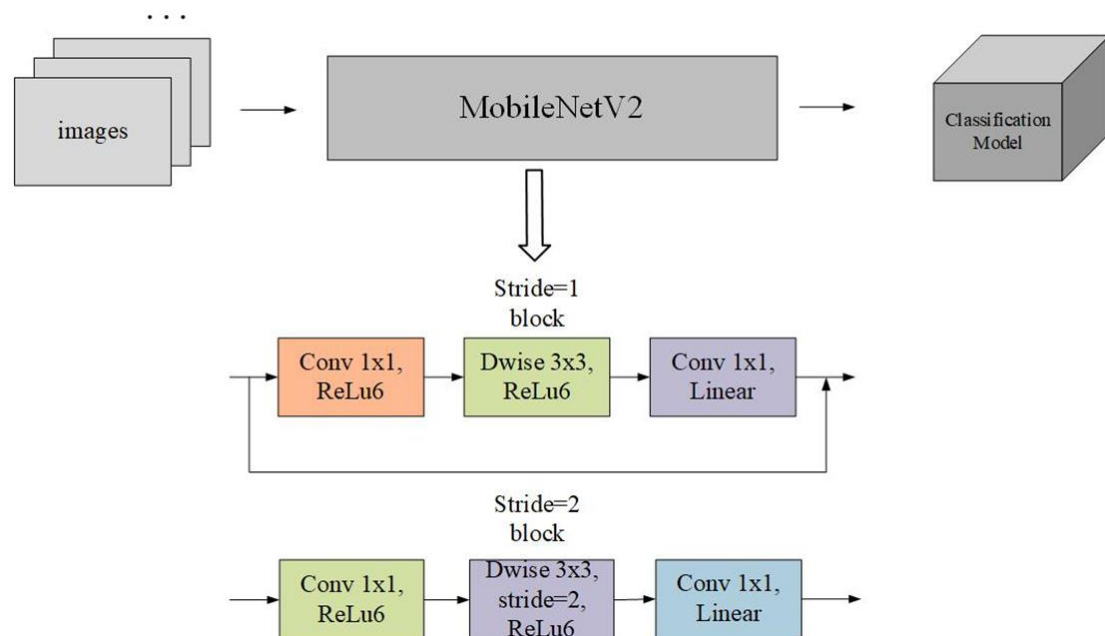
Problem 1: Image Classification

1. (2%) Draw the network architecture of method A or B.

- The graph should be brief and clear
- It would be fine to straight copy the figure from the paper

Method A: MobileNetV2(trained from scratch)

MobileNetV2 is a **CNN** architecture that seeks to perform well on **mobile devices**. It is based on an **inverted residual structure** where the residual connections are between the bottleneck layers. The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity. As a whole, the architecture of MobileNetV2 contains the initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers.



Ref:

1. <https://paperswithcode.com/method/mobilenetv2>
2. <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0282336>

2. (1%) Report accuracy of your models (both A, B) on the validation set.

	Method A: MobileNetV2(trained from scratch)	Method B: MobileNetV2(pretrained on cifar100)
Acc on the validation set	0.6476	0.9714

3. (2%) Report your implementation details of model A.

● Including but not limited to optimizer, loss function, cross validation method, lr scheduling

Training Details:	Method A: MobileNetV2(trained from scratch)
Data augmentation(only do totensor in validation&testing)	RandomCrop, RandomHorizontalFlip, ColorJitter, RandomRotation
Input size	No resize(32*32)
Total epochs/Patience/best epoch	200/20/58
Batch size	256
criterion	Cross entropy
Learning rate	0.0003
optimizer	Adam
Weight decay	1e-5
cross validation method	no
lr scheduling	no

4. (3%) Report your alternative model or method in B, and describe its difference from model A.

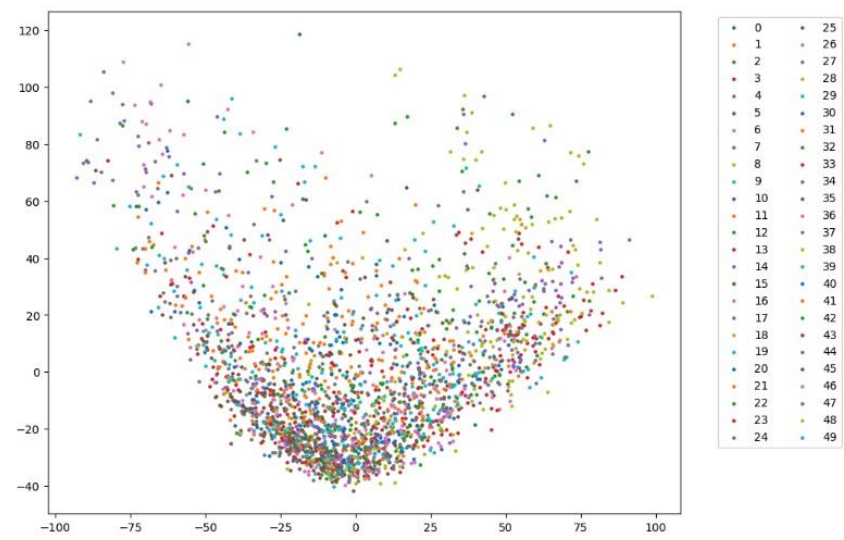
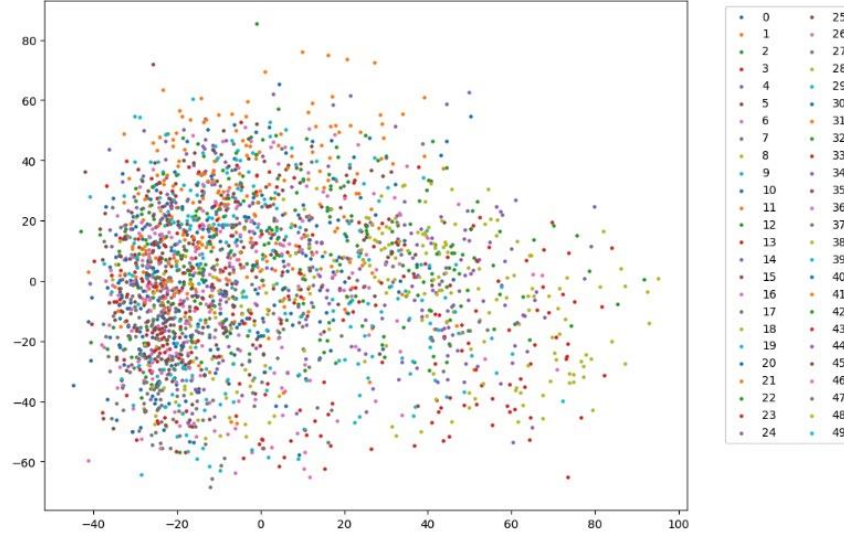
Method B: MobileNetV2(pretrained on cifar100)

The only difference is it is Method A pretrained on cifar100 (other training details are almost the same).

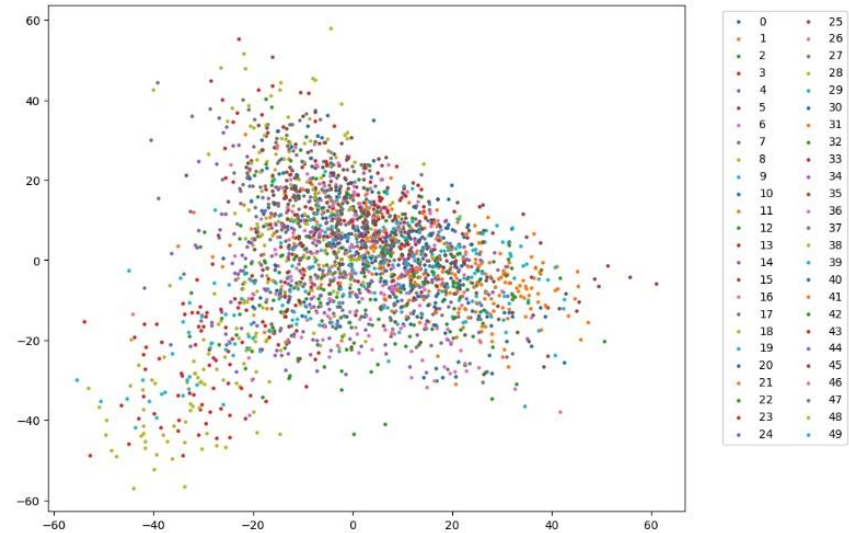
Pretrained weight from: <https://github.com/chenyaofu/pytorch-cifar-models/tree/master#model-zoo>

	Method B: MobileNetV2(pretrained on cifar100)
Acc on the validation set	0.9714
Training details	
Data augmentation(only do totensor in validation&testing)	RandomCrop, RandomHorizontalFlip,
Input size	No resize(32*32)
Total epochs/Patience/best epoch	150/15/1
Batch size	256
criterion	Cross entropy
Learning rate	0.0003
optimizer	Adam
Weight decay	1e-5
cross validation method	no
lr scheduling	no

5. (3%) 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.

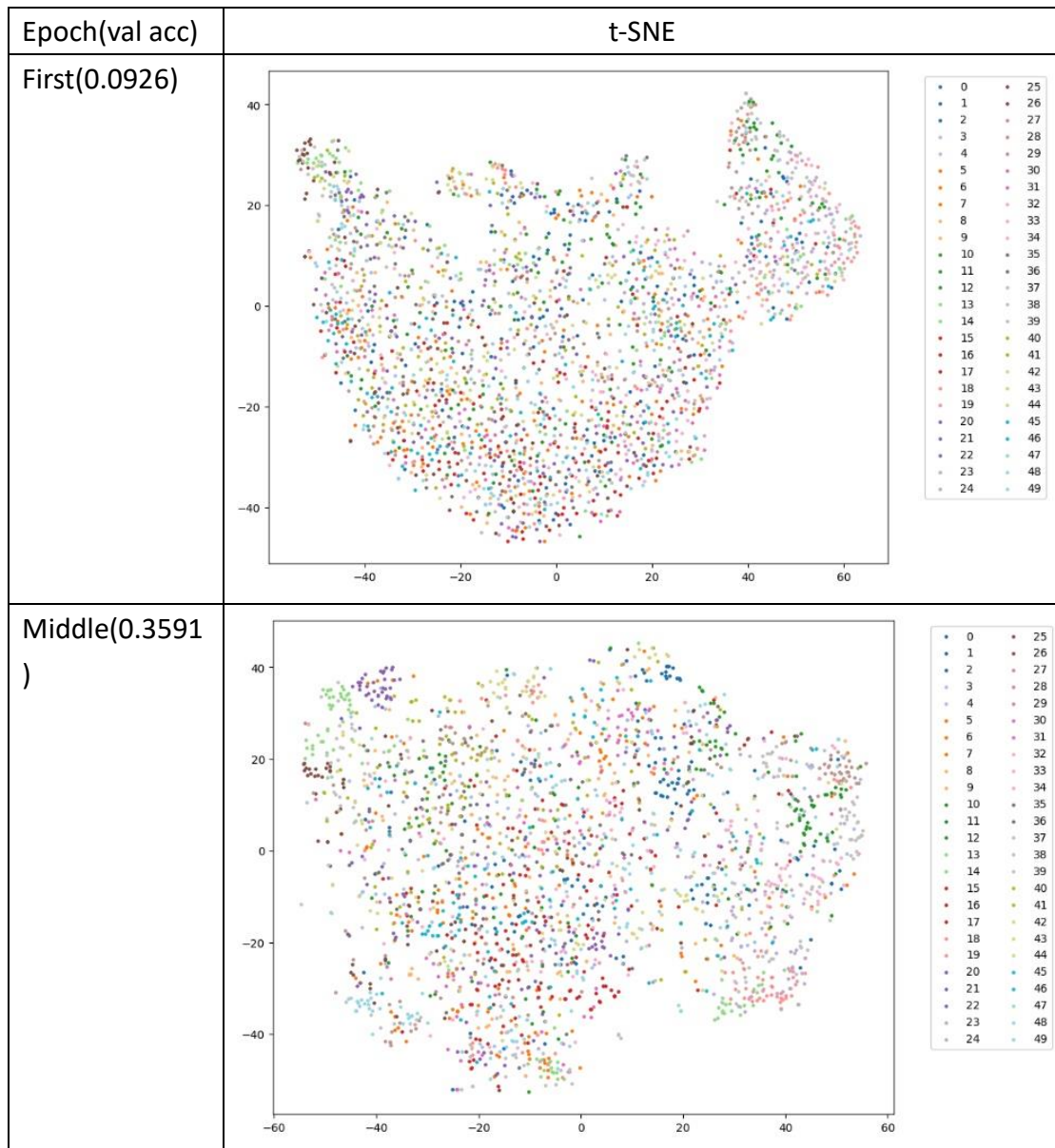
Epoch(val acc)	PCA
First(0.0926)	
Middle(0.3591))	

Last(0.6370)

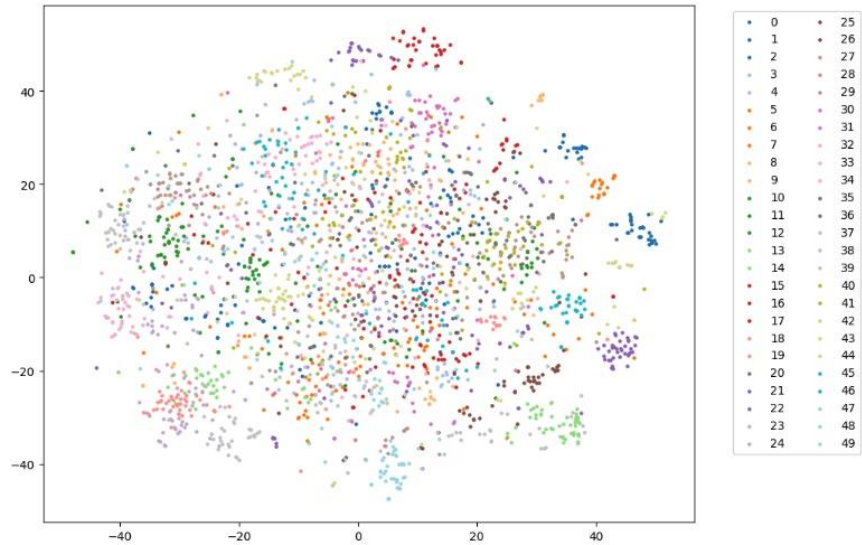


PCA is a **linear dimensionality reduction** technique. As a result, the visualizations for all three different stages (first, middle, last epochs) appear to be **inadequate for distinguishing samples with different labels**. The linear nature of PCA may not effectively capture the variations necessary to separate features belonging to different classes.

6. (4%) 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.



Last(0.6370)



T-SNE, being a dimensionality reduction technique, utilizes complex formulas to represent the relationship between high-dimensional and low-dimensional spaces. The visualization results from the final epoch of the model clearly demonstrate that inputs from different classes have separated into distinct clusters. This suggests that **t-SNE has effectively captured discriminative information from high-dimensional features.** Furthermore, **the visualizations from all three stages indicate a gradual separation of samples from different classes, indicating that the model is progressively learning distinctive features of samples from different categories.**

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

1. (5%) Describe the implementation details of your SSL method for pre-training the ResNet50 backbone. (Include but not limited to the name of the SSL method you used, data augmentation for SSL, learning rate schedule, optimizer, and batch size setting for this pre-training phase)

Due to the maximum usage GPU time on Kaggle, my SSL pre-training is divided into two stage(s1, s2).

SSL method name	Bootstrap Your Own Latent (BYOL) Ref: https://github.com/lucidrains/byol-pytorch/tree/master
data augmentation	s1, s2: Resize((128,128)), ToTensor()
data augmentation in BYOL	s1, s2: # default SimCLR augmentation ColorJitter, RandomGrayscale, RandomHorizontalFlip, GaussianBlur, RandomResizedCrop, Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
learning rate	s1, s2: 3e-4
learning rate schedule	s1: no learning rate schedule s2: torch.optim.lr_scheduler.ReduceLROnPlateau(opt, mode="min", factor=0.1, patience=10)
optimizer	Adam
batch size	128
Num of epochs(patience)	s1: 100(no patience setting) s2: 200(patience = 10)

2. (20%) 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.

The implementation details of fine-tuning (A~E are the same)	
data augmentation(only do Resize, Normalization(both same as the right) in validation&testing)	Resize((128,128)), RandomHorizontalFlip(), CenterCrop(128), Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
Batch size / optimizer / lr / wd / n_epochs / patience	256 / Adam / 3e-4 / 1e-5 / 150 / 15

Setting	Pre-training (Mini-ImageNet)	Fine-tuning (Office-Home dataset)	Validation accuracy (Office-Home dataset)
A	-	Train full model (backbone + classifier)	<u>0.2848</u>
B	w/ label (TAs have provided this backbone)	Train full model (backbone + classifier)	<u>0.4114</u>
C	w/o label (Your SSL pre-trained backbone)	Train full model (backbone + classifier)	<u>0.4551</u>
D	w/ label (TAs have provided this backbone)	Fix the backbone. Train classifier only	<u>0.3062</u>
E	w/o label (Your SSL pre-trained backbone)	Fix the backbone. Train classifier only	<u>0.3087</u>

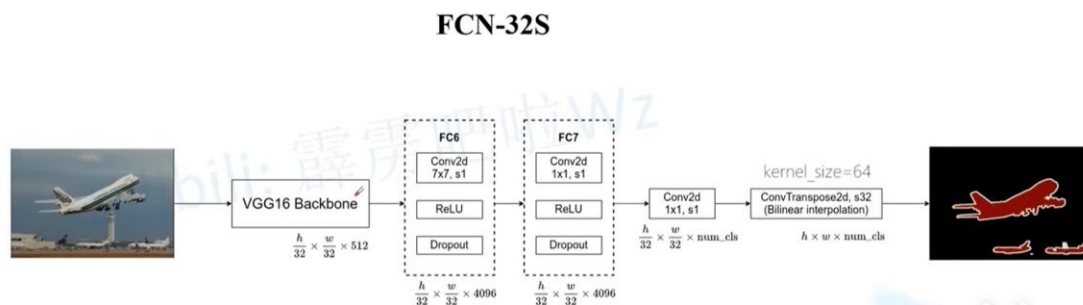
觀察如下:

1. 從 B, C 的 val acc 都遠比 A 好得知 -> 有做 pretrain (不論是 SL 或 SSL)後再去做下游任務比直接做下游任務的結果要好 -> 我認為 classifier 能做的事有限，影響結果的關鍵是 backbone 能否產生好的特徵，有比較好的結果是因為 fine tune 後 SSL or SL 的 backbone 比 A 的 backbone(train from scratch)有更好的特徵
2. 從 B 遠比 D 好與 C 遠比 E 好(都提升約 15%)得知 -> Fine-tune 整個模型遠比只 Fine-tune classifier 的結果好 -> backbone 若也能被 fine-tune 則能萃取出更符合下游任務的特徵，是使結果變好的關鍵(classifier 能做的是有限)
3. finetune backbone 則最後結果是 SSL 比 SL 好 -> SSL 得到的 pretrained weight 比 SL 好
4. 若 finetune backbone 則最後結果是 SSL 比 SL 好，只 finetune classifier 則兩者結果差不多 -> 證明兩個 pretrain weight 唯有在 fine tune backbone 後才能發揮各自的實力
5. 我認為 classifier 能做的事有限，從有 SSL or SL 的 backbone 但只 finetune classifier 的結果稍微比 A 好得知 -> A 的 backbone 產生的特徵和沒有 finetune 的 SSL or SL backbone 產生的特徵效果差不多 -> 再次驗證了 4. 的推論

Problem 3: Semantic Segmentation

1. (3%) Draw the network architecture of your VGG16-FCN32s model (model A).

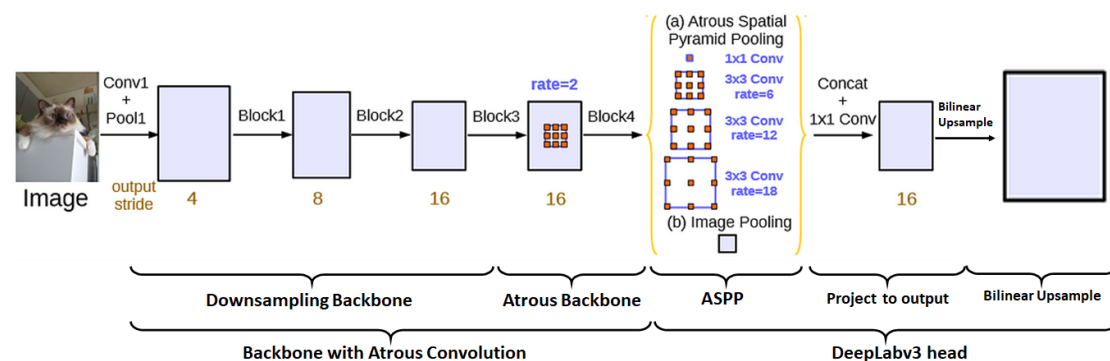
Model A: VGG16-FCN32s



Ref: https://www.bilibili.com/video/BV1J3411C7zd/?spm_id_from=333.337.search-card.all.click&vd_source=4e30bd473a2058d45e114f2ca6075e61

2. (3%) Draw the network architecture of the improved model (model B) and explain it differs from your VGG16-FCN32s model.

Model B: DEEPLABV3_MOBILENET_V3_LARGE



Ref:

- <https://arxiv.org/abs/1706.05587>
- <https://zhuanlan.zhihu.com/p/75333140>
- <https://medium.com/@itberrios6/deeplabv3-c0c8c93d25a4>

Differences from model A:







- FCN-32s 只考慮一張特徵圖，Deeplabv3 考慮了 **multi-scale** 的特徵圖來得到結果，故能得到更好的切割結果(能捕捉到細部特徵)。
- Deeplabv3 使用的是 **Atrous Spatial Pyramid Pooling (ASPP)**，其作法是併行的採用多個取樣率的 **Atrous Convolution** 來提取特徵，再將特徵融合，類似於空間金字塔結構。其中 **Atrous Convolution** 相較於原始的 **convolution** 的優點是：在不做 **pooling** 損失資訊和相同計算量下，加大了 **receptive field**，使每

個 filter 輸出都包含較大範圍的資訊。故與 FCN-32s(或甚至有用 multi-scale 特徵的 FCN-8s)比較，其產生的特徵圖包含更大範圍的資訊。

3. (1%) Report mIoUs of two models on the validation set.

	Model A: VGG16-FCN32s	Model B: DEEPLABV3_MOBILENET_V3_LARGE (pretrained on a subset of COCO, using only the 20 categories that are present in the Pascal VOC dataset)
mIoUs on the validation set	0.6257	0.7341

4. (3%) 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.

	0013_sat.jpg	0062_sat.jpg	0104_sat.jpg
First (mIoUs=0.5775)			
Middle (mIoUs=0.6537)			
Final (mIoUs=0.7166)	