Pre-trained Model

PIRL, POSTECH

Hanul Roh

Contents

- Transfer Learning
  - Create model (vgg-16)
  - Initialize parameters from pre-trained model
  - Training the model with CIFAR-10
- Semantic Segmentation using FCN

Transfer Learning

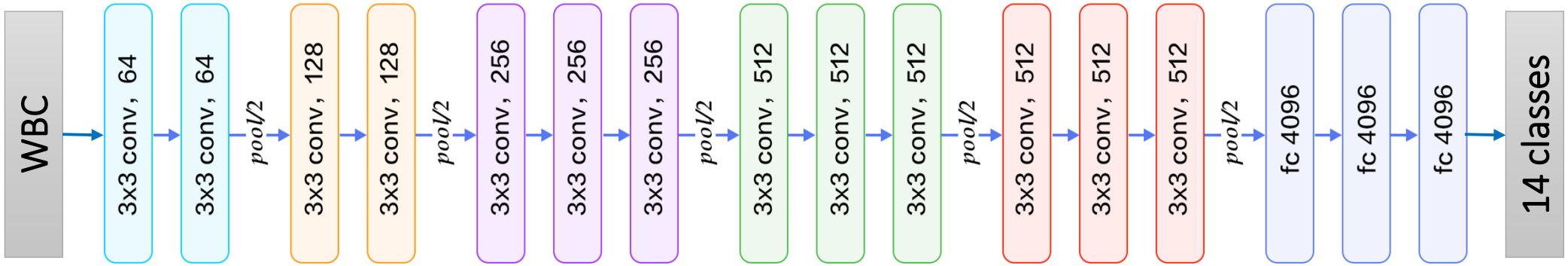
- Exploit the representation power learned from large scale dataset (e.g. ImageNet)

- Reason why…
  - Too few dataset
  - Too much time to train model from scratch
  - Overfitting problem

Transfer Learning

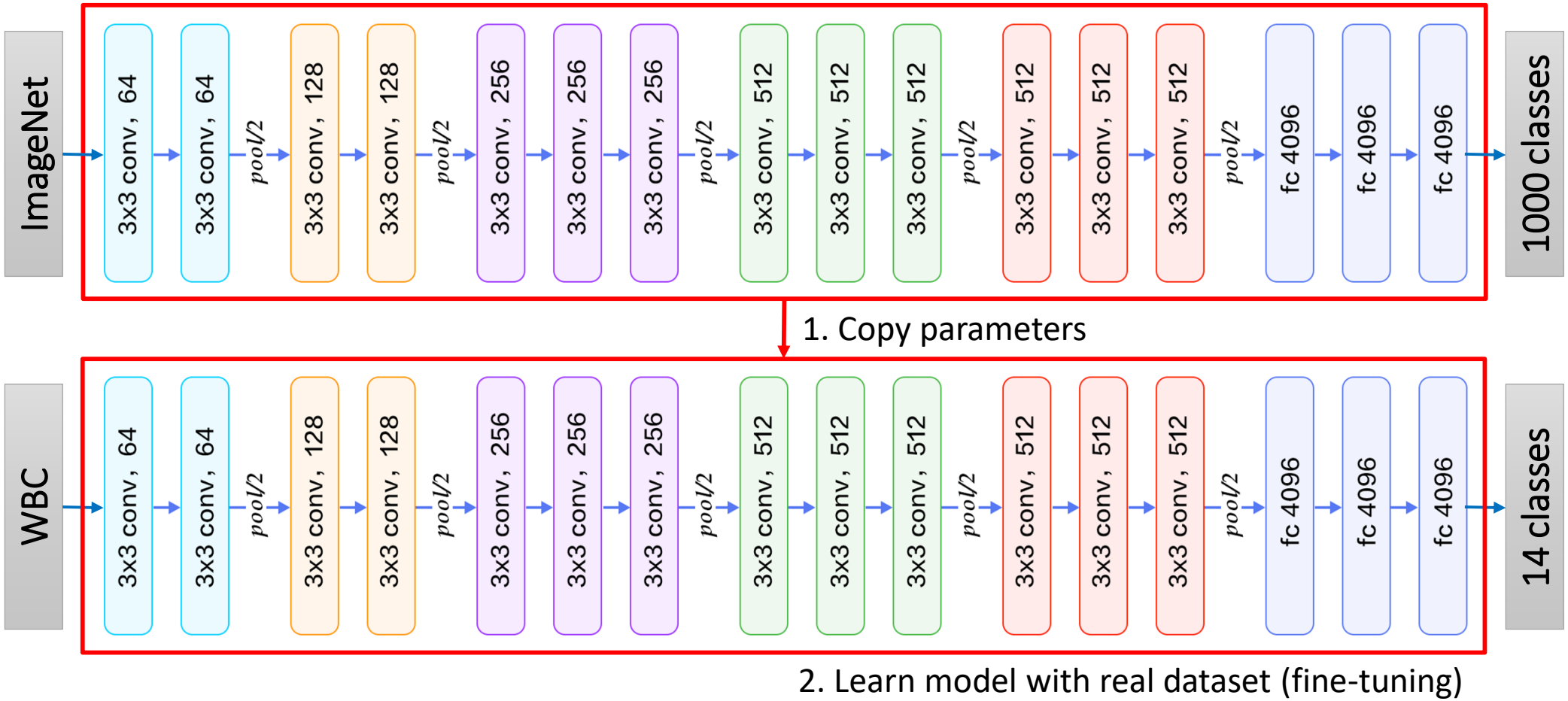
- 1000 categories, 1.2M training data, 100K test data (ILSVRC 2014)

Transfer Learning



- Imbalanced dataset (total 0.1K images)
  - Some class less than 200 images, the other more than 40000 images

Transfer Learning



Pre-trained Models

- How to use popular pre-trained models?
- In tensorflow, use TF-Slim library
  - AlexNet, VGGNet, ResNet, Inception, etc
  - https://github.com/tensorflow/models/tree/master/slim

Pre-trained Models

- Need two steps:
  - Construct a network equivalent to the pre-trained model with same name space in checkpoints from TF-Slim

Use tf.contrib.slim to construct pre-trained models (here, vgg-16)

```
from nets import vgg
with tf.contrib.slim.arg_scope(vgg.vgg_arg_scope()):
    logits, _ = vgg.vgg_16(images, num_classes=10, is_training=True)
```

* **vgg.py** includes construction functions for vgg models
  - defined in https://github.com/tensorflow/models/tree/master/slim/nets

* Or, you can define own function, but should follow the name space in the checkpoint

```
vgg_16/conv1/conv1_1/weights
vgg_16/conv1/conv1_1/biases
vgg_16/conv1/conv1_2/weights
vgg_16/conv1/conv1_2/biases
vgg_16/conv2/conv2_1/weights
vgg_16/conv2/conv2_1/biases
vgg_16/conv2/conv2_2/weights
vgg_16/conv2/conv2_2/biases
vgg_16/conv3/conv3_1/weights
vgg_16/conv3/conv3_1/biases
vgg_16/conv3/conv3_2/weights
vgg_16/conv3/conv3_2/biases
vgg_16/conv3/conv3_3/weights
vgg_16/conv3/conv3_3/biases
vgg_16/conv4/conv4_1/weights
vgg_16/conv4/conv4_1/biases
vgg_16/conv4/conv4_2/weights
vgg_16/conv4/conv4_2/biases
vgg_16/conv4/conv4_3/weights
vgg_16/conv4/conv4_3/biases
vgg_16/conv5/conv5_1/weights
vgg_16/conv5/conv5_1/biases
vgg_16/conv5/conv5_2/weights
vgg_16/conv5/conv5_2/biases
vgg_16/conv5/conv5_3/weights
vgg_16/conv5/conv5_3/biases
```

  - Select parameters to be copied and copy them using tf.train.Saver()

Pre-trained Models

- Need two steps:
  - Construct a network equivalent to the pre-trained model with same name space in checkpoints from TF-Slim
  - Select parameters to be copied and copy them using tf.train.Saver()

Select parameter variables

```
slim = tf.contrib.slim
exclude_layers = ['vgg_16/fc8']
variables_to_restore =
    slim.get_variables_to_restore(exclude=exclude_layers)
```

Restore the parameters

```
restorer = tf.train.Saver(variables_to_restorer)
sess = tf.Session()
restorer.restore(sess, save_path=checkpoint_path)
```

```
variables_to_restore = []
for var in tf.global_variables():
    excluded = False
    for exclusion in exclude_layers:
        if var.op.name.startswith(exclusion):
            excluded = True
            break
    if not excluded: variables_to_restore.append(var)
```

Exercise

- Download git from
  - https://github.com/mixcheck/cnntutorial
- Train vgg-16 using CIFAR-10 from scratch
- Train vgg-16 using CIFAR-10 from pre-trained model
  - Only fc8 layer
  - All layers
- Train ResNet-V1-50 from pre-trained model
  - https://github.com/tensorflow/models/tree/master/slim#Pretrained