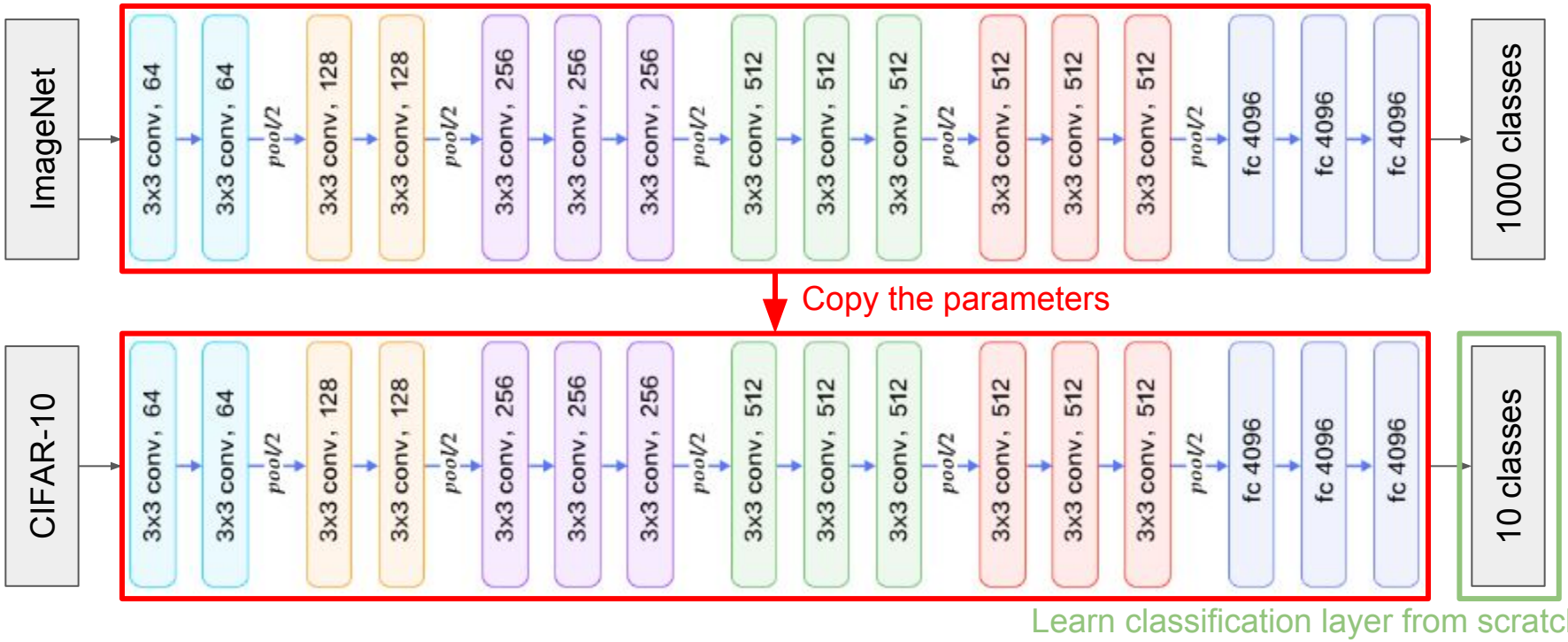
Pre-trained models

Transfer Learning

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



Transfer Learning

How to use popular pre-trained models?

In TF-Slim library,

- There is repository for pre-trained models like VGGNet, ResNet, Inception
- https://github.com/tensorflow/models/tree/master/slim

Transfer Learning

Need two steps:

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

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
```

Transfer Learning

Need two steps:

1. Construct a network equivalent to the pre-trained model with same name space in checkpoints from TF-Slim
2. **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)
```

Check the code pre_trained_models.ipynb

Exercises

1. Train the model in CIFAR-10 with pre-trained models
    a. For vggNet, need to change preprocessing function to perform
        i. Resizing images to 224x224 size
        ii. Converting RGB to BGR
        iii. Subtracting mean value of BGR (103.939, 116.779, 123.68)
    b. Learning from scratch
    c. Learning by transferring the weights
        i. Only learning fc8 layer
        ii. Learning all layers
    d. Different pre-trained model, not vgg-16, e.g. ResNet or Inception models