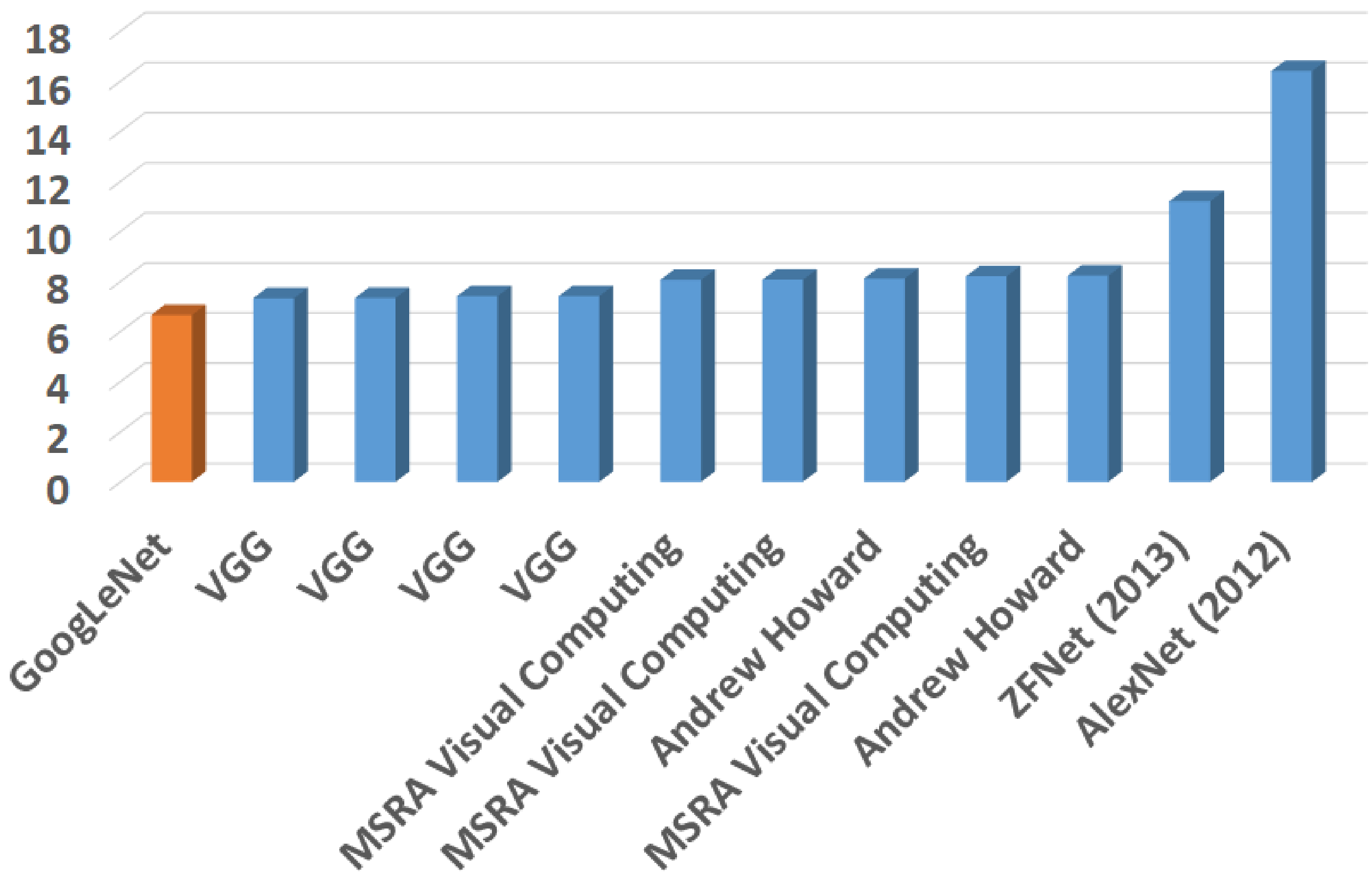
VGGNet

Kyeong Hwan Moon

01 GoogleNet

Introduction

Error Rate in ILSVRC 2014 (%)



01 Dataset

Inspection

n02097047 (196)

n01682714 (40)

n03134739 (522)

n04254777 (806)

n02859443 (449)

n02096177 (192)

n02107683 (239)

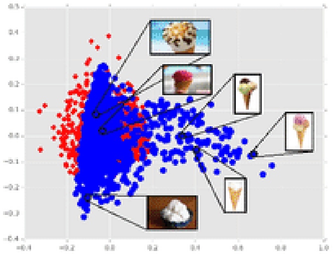
n01443537 (1)

n02264363 (318)

Icecream - ILSVRC12 (IN)

Icecream - WIN

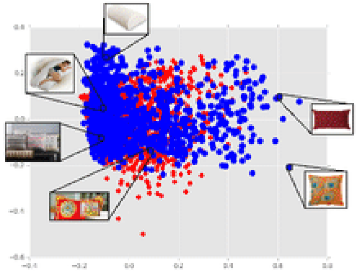
Icecream - WINC


ILSVR red, WINC blue

Pillow - ILSVRC12 (IN)

Pillow - WIN

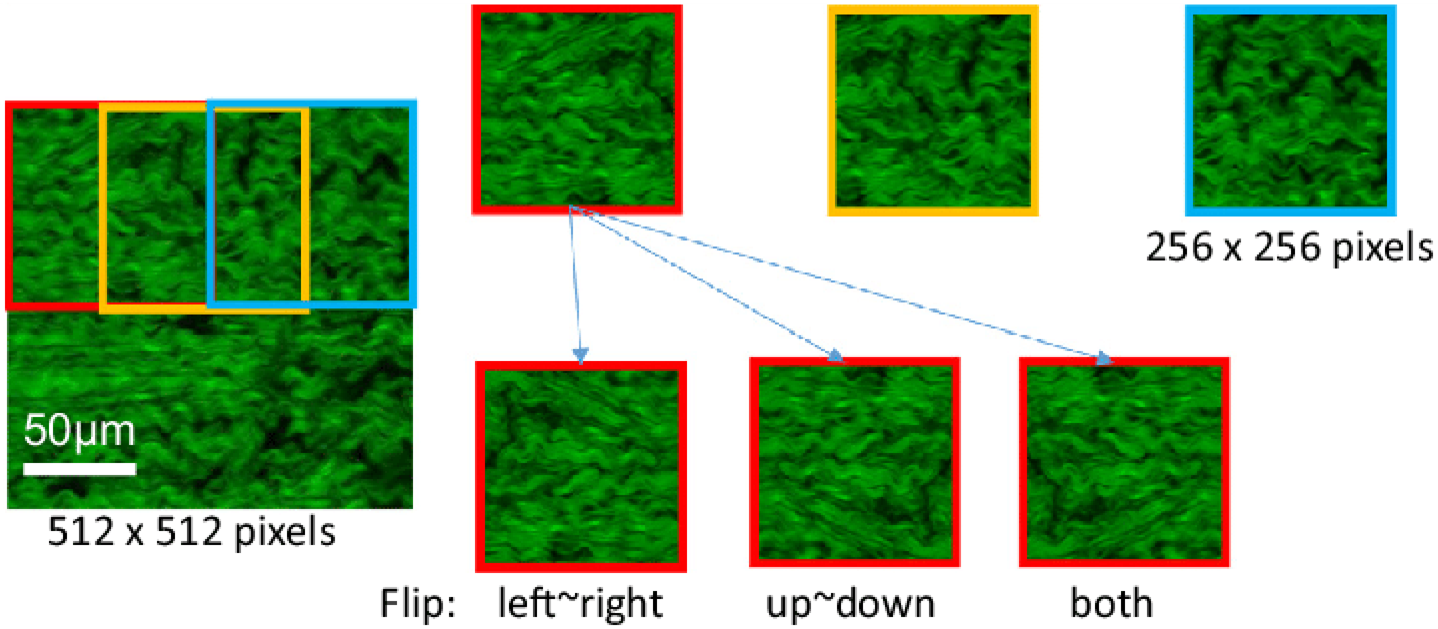
Pillow - WINC


ILSVR red, WINC blue

01 Preprocessing

Data Augmentation

- Extracting random 224×224 patches
- Subtracted the mean RGB value, computed on the training set

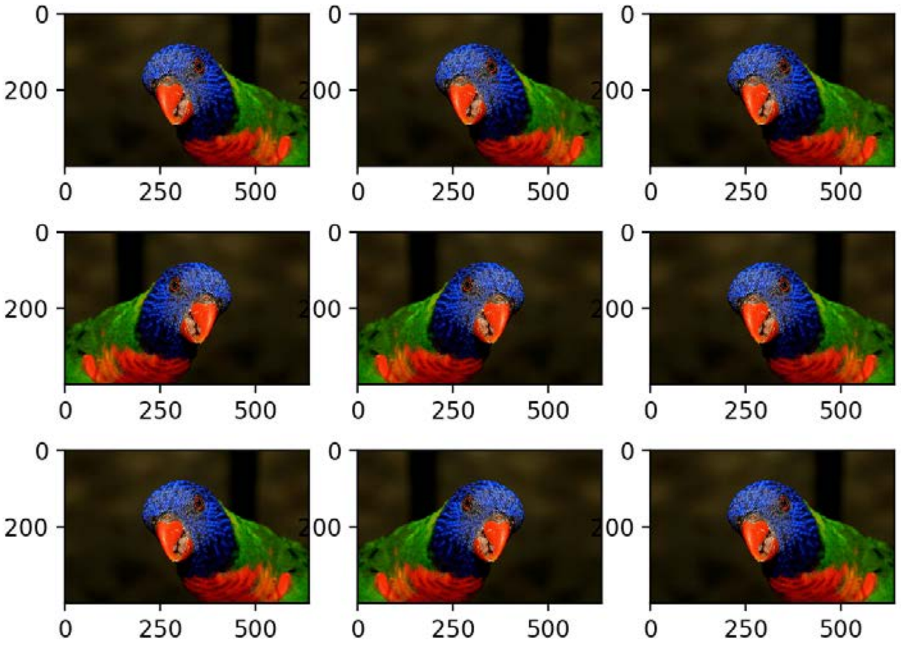


** Size of image is different compared with original AelxNet data augmentation

01 Preprocessing

Data Augmentation

horizontal reflections

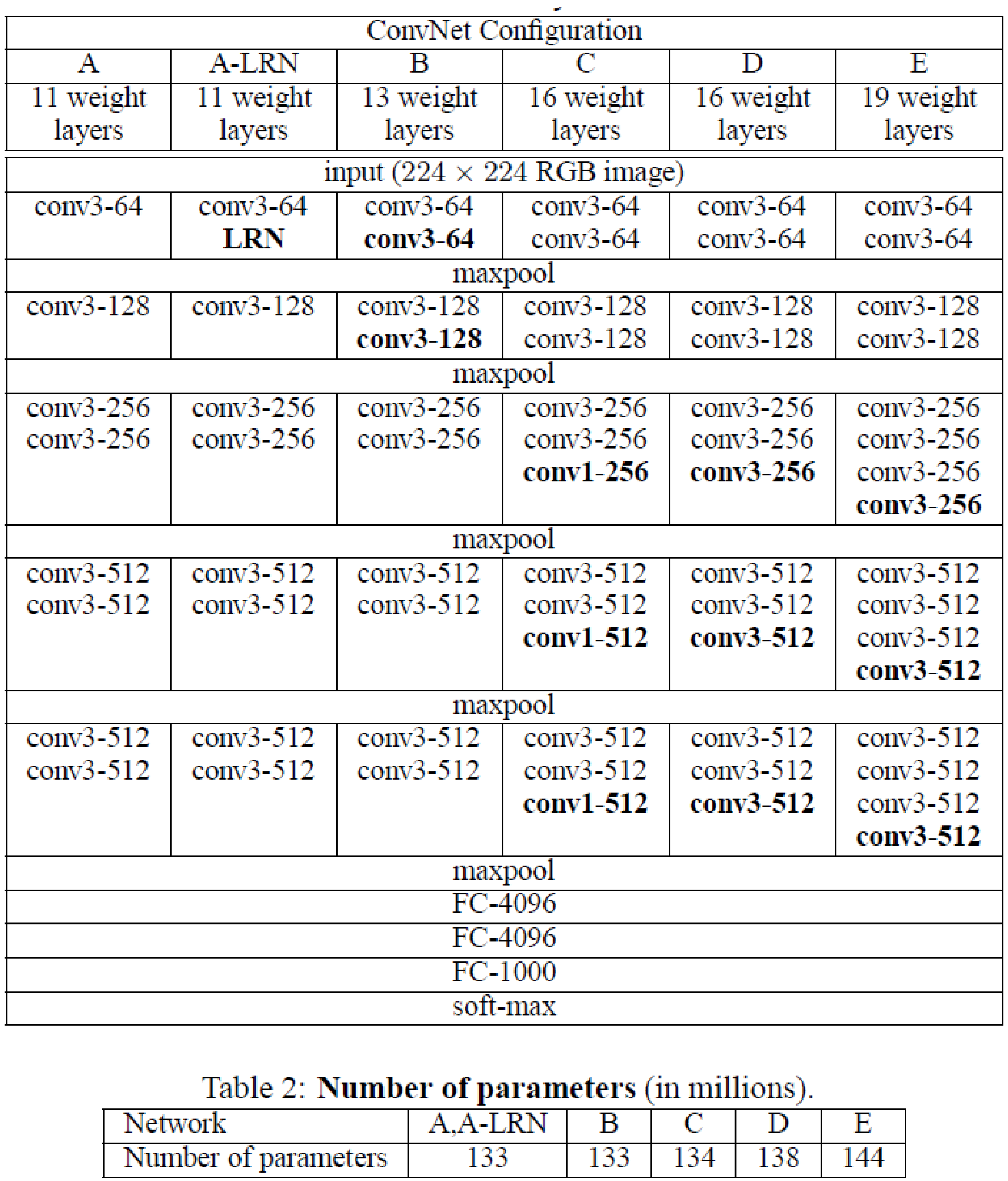


Color jittering

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$

01 VGGNet

Table



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

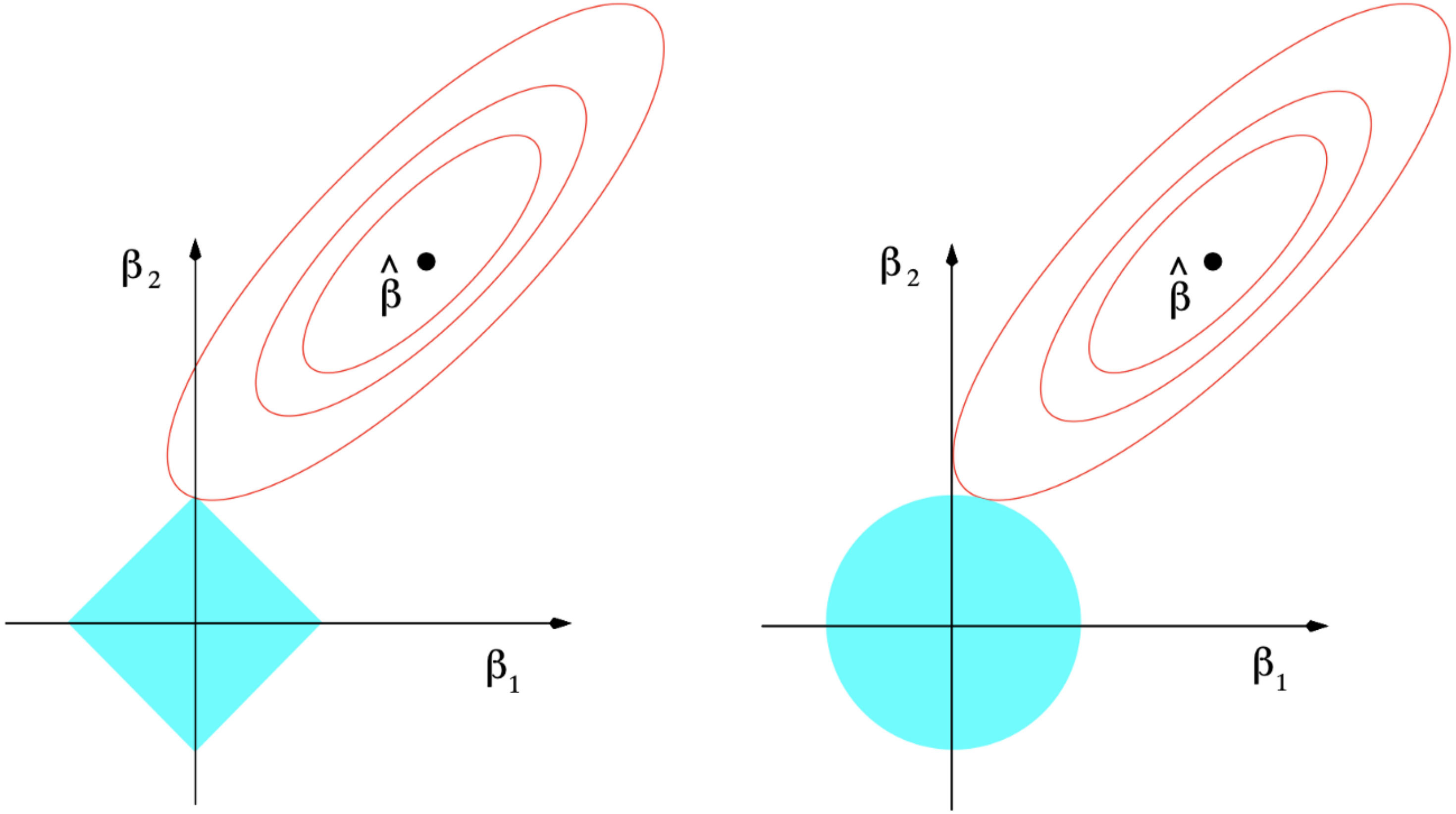
Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

01

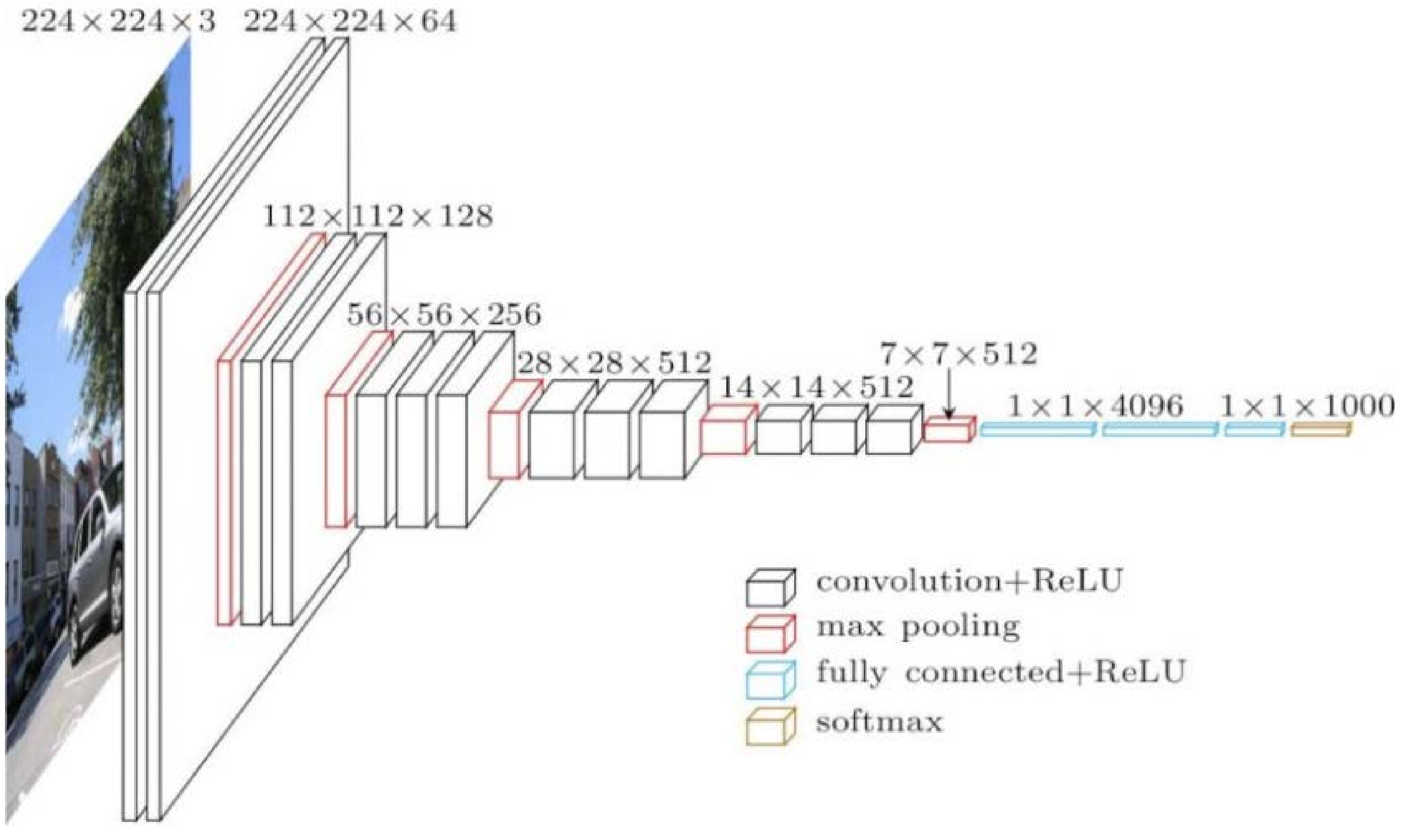
L2

Lasso and Ridge

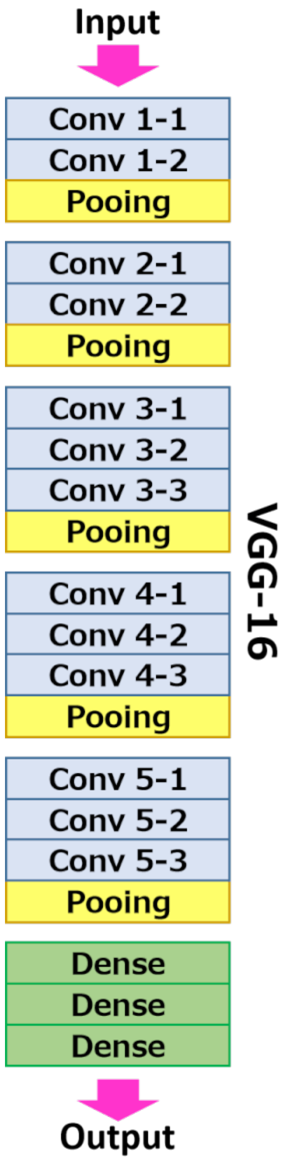


01 VGGNet

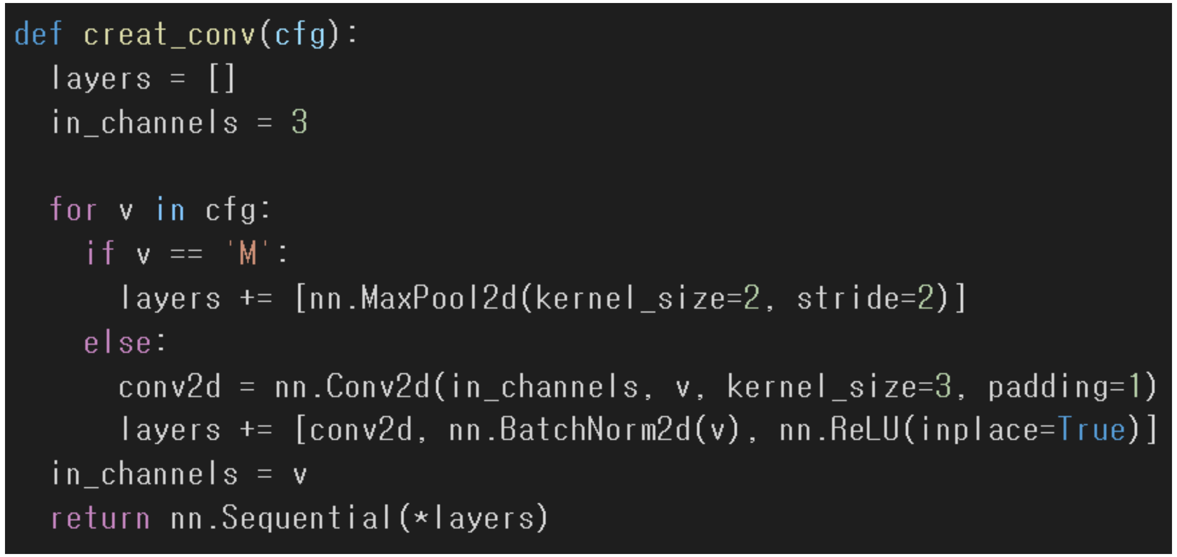
VGGNet



VGG16 Architecture

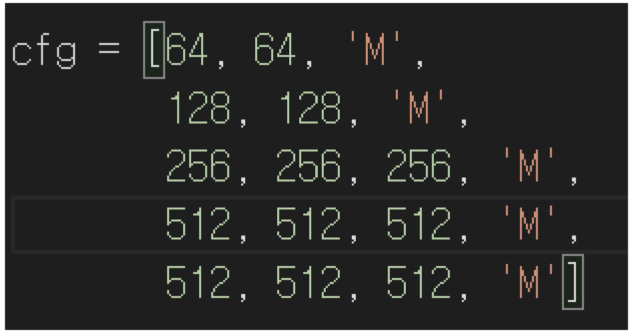


01 Code



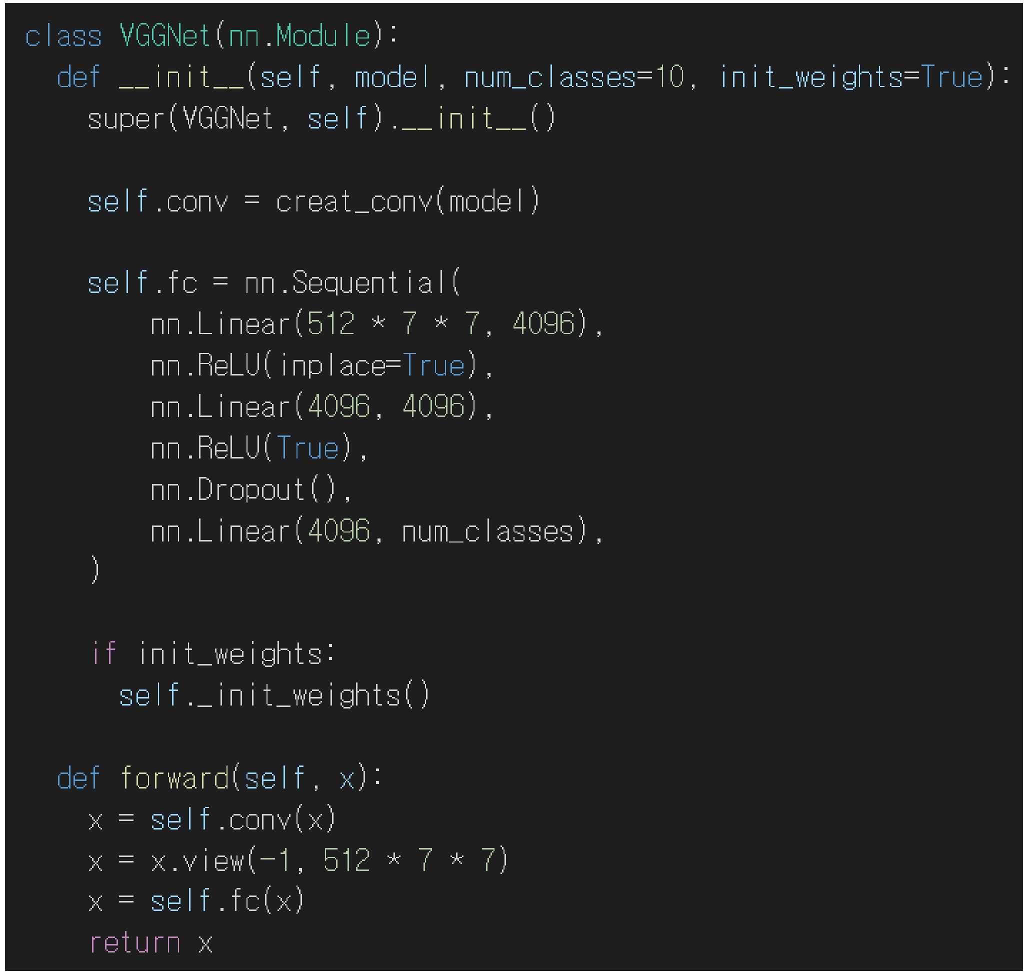
```
def creat_conv(cfg):
  layers = []
  in_channels = 3

  for v in cfg:
    if v == 'M':
      layers += [nn.MaxPool2d(kernel_size=2, stride=2)]
    else:
      conv2d = nn.Conv2d(in_channels, v, kernel_size=3, padding=1)
      layers += [conv2d, nn.BatchNorm2d(v), nn.ReLU(inplace=True)]
  in_channels = v
  return nn.Sequential(*layers)
```



```
cfg = [64, 64, 'M',
       128, 128, 'M',
       256, 256, 256, 'M',
       512, 512, 512, 'M',
       512, 512, 512, 'M']
```

01 Code



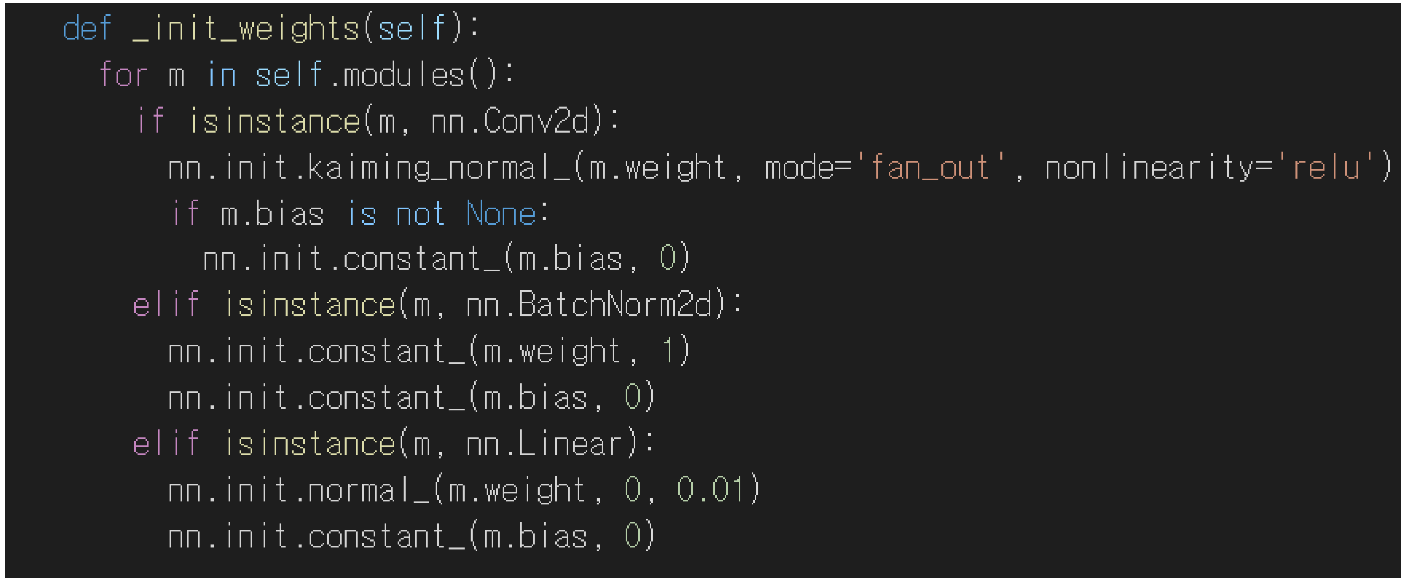
```python
class VGGNet(nn.Module):
  def __init__(self, model, num_classes=10, init_weights=True):
    super(VGGNet, self).__init__()

    self.conv = creat_conv(model)

    self.fc = nn.Sequential(
        nn.Linear(512 * 7 * 7, 4096),
        nn.ReLU(inplace=True),
        nn.Linear(4096, 4096),
        nn.ReLU(True),
        nn.Dropout(),
        nn.Linear(4096, num_classes),
    )

    if init_weights:
      self._init_weights()

  def forward(self, x):
    x = self.conv(x)
    x = x.view(-1, 512 * 7 * 7)
    x = self.fc(x)
    return x
```



```python
def _init_weights(self):
  for m in self.modules():
    if isinstance(m, nn.Conv2d):
      nn.init.kaiming_normal_(m.weight, mode='fan_out', nonlinearity='relu')
      if m.bias is not None:
        nn.init.constant_(m.bias, 0)
    elif isinstance(m, nn.BatchNorm2d):
      nn.init.constant_(m.weight, 1)
      nn.init.constant_(m.bias, 0)
    elif isinstance(m, nn.Linear):
      nn.init.normal_(m.weight, 0, 0.01)
      nn.init.constant_(m.bias, 0)
```

Thank You