Chapter7

April 29, 2019

D2L Textbook Solution

```
In [2]: import d2l
        import mxnet as mx
        from mxnet import autograd, gluon, init, nd
        from mxnet.gluon import loss as gloss, data as gdata, nn

        import time

        import numpy as np

        from IPython.core.interactiveshell import InteractiveShell
        InteractiveShell.ast_node_interactivity = "all"
```

0.0.1 Chapter 7

7.1 (AlexNet) Exercises

1. **Try increasing the number of epochs. Compared with LeNet, how are the results different? Why?**
2. **AlexNet may be too complex for the Fashion-MNIST data set.**
   a. **Try to simplify the model to make the training faster, while ensuring that the accuracy does not drop significantly.**
   b. **Can you design a better model that works directly on 28 x 28 images.**
3. **Modify the batch size, and observe the changes in accuracy and GPU memory.**
4. **Rooflines**
   a. **What is the dominant part for the memory footprint of AlexNet?**

   Dense1 - 26 millions of parameters

   b. **What is the dominant part for computation in AlexNet?**

   Conv2 - 16 millions of FLOP

   c. **How about memory bandwidth when computing the results?**
5. **Apply dropout and ReLU to LeNet5. Does it improve? How about preprocessing?**

7.2 (VGG) Exercises

1. **When printing out the dimensions of the layers we only saw 8 results rather than 11. Where did the remaining 3 layer informations go?**

   There are 3 pairs of CNN (3 vgg blocks) has the same shape

2. **Compared with AlexNet, VGG is much slower in terms of computation, and it also needs more GPU memory. Try to analyze the reasons for this.**

   As shown in the summary results below, VGG needs to train much more parameters than AlexNet.

```
In [5]: def vgg_block(num_convs, num_channels):
            blk = nn.Sequential()
            for _ in range(num_convs):
                blk.add(nn.Conv2D(num_channels, kernel_size=3,
                                  padding=1, activation='relu'))
            blk.add(nn.MaxPool2D(pool_size=2, strides=2))
            return blk

        def VGG11(conv_arch):
            net = nn.Sequential()
            # The convolutional layer part
            for (num_convs, num_channels) in conv_arch:
                net.add(vgg_block(num_convs, num_channels))
            # The fully connected layer part
            net.add(nn.Dense(4096, activation='relu'), nn.Dropout(0.5),
                    nn.Dense(4096, activation='relu'), nn.Dropout(0.5),
                    nn.Dense(10))
            return net

        conv_arch = ((1, 64), (1, 128), (2, 256), (2, 512), (2, 512))
        vgg = VGG11(conv_arch)
        vgg.initialize()
        X = nd.random.uniform(shape=(1, 1, 224, 224))
        for blk in vgg:
            X = blk(X)
            print(blk.name, 'output shape:\t', X.shape)
        vgg.summary(mx.nd.random.uniform(shape=(1, 1, 244, 244)))
```

```
sequential2 output shape:        (1, 64, 112, 112)
sequential3 output shape:        (1, 128, 56, 56)
sequential4 output shape:        (1, 256, 28, 28)
sequential5 output shape:        (1, 512, 14, 14)
sequential6 output shape:        (1, 512, 7, 7)
dense1 output shape:         (1, 4096)
dropout0 output shape:         (1, 4096)
dense2 output shape:         (1, 4096)
dropout1 output shape:         (1, 4096)
```

```
dense3 output shape:         (1, 10)
-------------------------------------------------------------------------------
        Layer (type)                               Output Shape         Param #
===============================================================================
               Input                            (1, 1, 244, 244)              0
        Activation-1                           (1, 64, 244, 244)              0
            Conv2D-2                           (1, 64, 244, 244)            640
         MaxPool2D-3                           (1, 64, 122, 122)              0
        Activation-4                          (1, 128, 122, 122)              0
            Conv2D-5                          (1, 128, 122, 122)          73856
         MaxPool2D-6                            (1, 128, 61, 61)              0
        Activation-7                            (1, 256, 61, 61)              0
            Conv2D-8                            (1, 256, 61, 61)         295168
        Activation-9                            (1, 256, 61, 61)              0
           Conv2D-10                            (1, 256, 61, 61)         590080
        MaxPool2D-11                            (1, 256, 30, 30)              0
       Activation-12                            (1, 512, 30, 30)              0
           Conv2D-13                            (1, 512, 30, 30)        1180160
       Activation-14                            (1, 512, 30, 30)              0
           Conv2D-15                            (1, 512, 30, 30)        2359808
        MaxPool2D-16                            (1, 512, 15, 15)              0
       Activation-17                            (1, 512, 15, 15)              0
           Conv2D-18                            (1, 512, 15, 15)        2359808
       Activation-19                            (1, 512, 15, 15)              0
           Conv2D-20                            (1, 512, 15, 15)        2359808
        MaxPool2D-21                              (1, 512, 7, 7)              0
       Activation-22                                   (1, 4096)              0
            Dense-23                                   (1, 4096)      102764544
          Dropout-24                                   (1, 4096)              0
       Activation-25                                   (1, 4096)              0
            Dense-26                                   (1, 4096)       16781312
          Dropout-27                                   (1, 4096)              0
            Dense-28                                     (1, 10)          40970
===============================================================================
Parameters in forward computation graph, duplicate included
   Total params: 128806154
   Trainable params: 128806154
   Non-trainable params: 0
Shared params in forward computation graph: 0
Unique parameters in model: 128806154
-------------------------------------------------------------------------------
```

3. **Try to change the height and width of the images in Fashion-MNIST from 224 to 96. What influence does this have on the experiments?**

   As it shows below, there will be 45M params in the model with resized images, compared with 129M params in the origin $224 \times 224 \times 3$ size images.

```
In [7]: vgg11 = VGG11(conv_arch)
        vgg11.initialize()
        vgg11.summary(mx.nd.random.uniform(shape=(1, 1, 96, 96)))
```

```
--------------------------------------------------------------------------------
        Layer (type)                                Output Shape         Param #
================================================================================
               Input                               (1, 1, 96, 96)              0
        Activation-1                     <Symbol conv65_relu_fwd>              0
        Activation-2                              (1, 64, 96, 96)              0
            Conv2D-3                              (1, 64, 96, 96)            640
         MaxPool2D-4                              (1, 64, 48, 48)              0
        Activation-5                     <Symbol conv66_relu_fwd>              0
        Activation-6                             (1, 128, 48, 48)              0
            Conv2D-7                             (1, 128, 48, 48)          73856
         MaxPool2D-8                             (1, 128, 24, 24)              0
        Activation-9                     <Symbol conv67_relu_fwd>              0
       Activation-10                             (1, 256, 24, 24)              0
           Conv2D-11                             (1, 256, 24, 24)         295168
       Activation-12                     <Symbol conv68_relu_fwd>              0
       Activation-13                             (1, 256, 24, 24)              0
           Conv2D-14                             (1, 256, 24, 24)         590080
        MaxPool2D-15                             (1, 256, 12, 12)              0
       Activation-16                     <Symbol conv69_relu_fwd>              0
       Activation-17                             (1, 512, 12, 12)              0
           Conv2D-18                             (1, 512, 12, 12)        1180160
       Activation-19                     <Symbol conv70_relu_fwd>              0
       Activation-20                             (1, 512, 12, 12)              0
           Conv2D-21                             (1, 512, 12, 12)        2359808
        MaxPool2D-22                               (1, 512, 6, 6)              0
       Activation-23                     <Symbol conv71_relu_fwd>              0
       Activation-24                               (1, 512, 6, 6)              0
           Conv2D-25                               (1, 512, 6, 6)        2359808
       Activation-26                     <Symbol conv72_relu_fwd>              0
       Activation-27                               (1, 512, 6, 6)              0
           Conv2D-28                               (1, 512, 6, 6)        2359808
        MaxPool2D-29                               (1, 512, 3, 3)              0
       Activation-30                     <Symbol dense4_relu_fwd>              0
       Activation-31                                    (1, 4096)              0
            Dense-32                                    (1, 4096)       18878464
          Dropout-33                                    (1, 4096)              0
       Activation-34                     <Symbol dense5_relu_fwd>              0
       Activation-35                                    (1, 4096)              0
            Dense-36                                    (1, 4096)       16781312
          Dropout-37                                    (1, 4096)              0
            Dense-38                                      (1, 10)          40970
================================================================================
Parameters in forward computation graph, duplicate included
```

```
   Total params: 44920074
   Trainable params: 44920074
   Non-trainable params: 0
Shared params in forward computation graph: 0
Unique parameters in model: 44920074
--------------------------------------------------------------------------------
```

4. **Refer to Table 1 in the original VGG Paper to construct other common models, such as VGG-16 or VGG-19.**

7.3 (NiN) Exercises

1. **Tune the hyper-parameters to improve the classification accuracy.**

   See STAT157 Homework4 for details.

2. **Why are there two 1 x 1 convolutional layers in the NiN block? Remove one of them, and then observe and analyze the experimental phenomena.**

3. **Calculate the resource usage for NiN:**

   a. **What is the number of parameters?** ~2M.
   b. **What is the amount of computation?**
   c. **What is the amount of memory needed during training?**
   d. **What is the amount of memory needed during inference?**

```
In [8]: def nin_block(num_channels, kernel_size, strides, padding):
            blk = nn.Sequential()
            blk.add(nn.Conv2D(num_channels, kernel_size, strides, padding, activation='relu'),
                    nn.Conv2D(num_channels, kernel_size=1, activation='relu'),
                    nn.Conv2D(num_channels, kernel_size=1, activation='relu'))
            return blk

        NiNnet = nn.Sequential()
        NiNnet.add(nin_block(96, kernel_size=11, strides=4, padding=0),
                nn.MaxPool2D(pool_size=3, strides=2),
                nin_block(256, kernel_size=5, strides=1, padding=2), nn.MaxPool2D(pool_size=3,
                nin_block(384, kernel_size=3, strides=1, padding=1), nn.MaxPool2D(pool_size=3,
                nn.Dropout(0.5),
                # There are 10 label classes
                nin_block(10, kernel_size=3, strides=1, padding=1),
                # The global average pooling layer automatically sets the window shape # to th
                nn.GlobalAvgPool2D(),
                # Transform the four-dimensional output into two-dimensional output
                # with a shape of (batch size, 10)
                nn.Flatten())

        ## architechture as Textbok
```

```python
X = nd.random.uniform(shape=(1, 1, 224, 224))
NiNnet.initialize()
for layer in NiNnet:
    X = layer(X)
    print(layer.name, 'output shape:\t', X.shape)

NiNnet.summary(mx.nd.random.uniform(shape=(1, 1, 244, 244)))
```

```
sequential14 output shape:       (1, 96, 54, 54)
pool24 output shape:       (1, 96, 26, 26)
sequential15 output shape:       (1, 256, 26, 26)
pool25 output shape:       (1, 256, 12, 12)
sequential16 output shape:       (1, 384, 12, 12)
pool26 output shape:       (1, 384, 5, 5)
dropout4 output shape:       (1, 384, 5, 5)
sequential17 output shape:       (1, 10, 5, 5)
pool27 output shape:       (1, 10, 1, 1)
flatten0 output shape:       (1, 10)
--------------------------------------------------------------------------------
        Layer (type)                                Output Shape         Param #
================================================================================
               Input                            (1, 1, 244, 244)               0
        Activation-1                             (1, 96, 59, 59)               0
            Conv2D-2                             (1, 96, 59, 59)           11712
        Activation-3                             (1, 96, 59, 59)               0
            Conv2D-4                             (1, 96, 59, 59)            9312
        Activation-5                             (1, 96, 59, 59)               0
            Conv2D-6                             (1, 96, 59, 59)            9312
         MaxPool2D-7                             (1, 96, 29, 29)               0
        Activation-8                            (1, 256, 29, 29)               0
            Conv2D-9                            (1, 256, 29, 29)          614656
       Activation-10                            (1, 256, 29, 29)               0
           Conv2D-11                            (1, 256, 29, 29)           65792
       Activation-12                            (1, 256, 29, 29)               0
           Conv2D-13                            (1, 256, 29, 29)           65792
        MaxPool2D-14                            (1, 256, 14, 14)               0
       Activation-15                            (1, 384, 14, 14)               0
           Conv2D-16                            (1, 384, 14, 14)          885120
       Activation-17                            (1, 384, 14, 14)               0
           Conv2D-18                            (1, 384, 14, 14)          147840
       Activation-19                            (1, 384, 14, 14)               0
           Conv2D-20                            (1, 384, 14, 14)          147840
        MaxPool2D-21                              (1, 384, 6, 6)               0
          Dropout-22                              (1, 384, 6, 6)               0
       Activation-23                               (1, 10, 6, 6)               0
           Conv2D-24                               (1, 10, 6, 6)           34570
       Activation-25                               (1, 10, 6, 6)               0
           Conv2D-26                               (1, 10, 6, 6)             110
       Activation-27                                (1, 10, 6, 6)                0
           Conv2D-28                                (1, 10, 6, 6)              110
  GlobalAvgPool2D-29                                (1, 10, 1, 1)                0
          Flatten-30                                      (1, 10)                0
================================================================================
Parameters in forward computation graph, duplicate included
   Total params: 1992166
   Trainable params: 1992166
   Non-trainable params: 0
Shared params in forward computation graph: 0
Unique parameters in model: 1992166
--------------------------------------------------------------------------------
```

4. **What are possible problems with reducing the 384 x 5 x 5 representation to a 10 x 5 x 5 representation in one step?**

   Loss too much information at one dimension in one step.

7.4 (GoogLeNet) Exercises

1. **There are several iterations of GoogLeNet. Try to implement and run them. Some of them include the following:**

   a. **Add a batch normalization layer, as described later in this chapter.**
   b. **Make adjustments to the Inception block.**
   c. **Use "label smoothing" for model regularization.**

   "label smoothing" has the advantage of preventing the pursuit of hard probabilities without discouraging correct classification.

   Check `smooth_alpha` in `mxnet.ndarray.Softmax`.

   https://mxnet.incubator.apache.org/api/python/ndarray/ndarray.html?highlight=softmaxoutput#mxne

   d. **Include it in the residual connection, as described later in this chapter.**

```
In [4]: class Inception(nn.Block):
            # c1 - c4 are the number of output channels for each layer in the path
            def __init__(self, c1, c2, c3, c4, **kwargs):
                super(Inception, self).__init__(**kwargs)
                self.p1_1 = nn.Conv2D(c1, kernel_size=1, activation='relu')
                self.p2_1 = nn.Conv2D(c2[0], kernel_size=1, activation='relu')
                self.p2_2 = nn.Conv2D(c2[1], kernel_size=3, padding=1, activation='relu')
                self.p3_1 = nn.Conv2D(c3[0], kernel_size=1, activation='relu')
                self.p3_2 = nn.Conv2D(c3[1], kernel_size=5, padding=2,
                                      activation='relu')
                self.p4_1 = nn.MaxPool2D(pool_size=3, strides=1, padding=1)
                self.p4_2 = nn.Conv2D(c4, kernel_size=1, activation='relu')
```

```
    def forward(self, x):
        p1 = self.p1_1(x)
        p2 = self.p2_2(self.p2_1(x))
        p3 = self.p3_2(self.p3_1(x))
        p4 = self.p4_2(self.p4_1(x))
        # Concatenate the outputs on the channel dimension
        return nd.concat(p1, p2, p3, p4, dim=1)

def GoogLeNet():
    net = nn.Sequential()
    net.add(nn.Conv2D(64, kernel_size=7, strides=2, padding=3, activation='relu'),
           nn.MaxPool2D(pool_size=3, strides=2, padding=1))
    net.add(nn.Conv2D(64, kernel_size=1, activation='relu'),
           nn.Conv2D(192, kernel_size=3, padding=1, activation='relu'),
           nn.MaxPool2D(pool_size=3, strides=2, padding=1))
    net.add(Inception(64, (96, 128), (16, 32), 32),
           Inception(128, (128, 192), (32, 96), 64),
           nn.MaxPool2D(pool_size=3, strides=2, padding=1))
    net.add(Inception(192, (96, 208), (16, 48), 64),
           Inception(160, (112, 224), (24, 64), 64),
           Inception(128, (128, 256), (24, 64), 64),
           Inception(112, (144, 288), (32, 64), 64),
           Inception(256, (160, 320), (32, 128), 128),
           nn.MaxPool2D(pool_size=3, strides=2, padding=1))
    net.add(Inception(256, (160, 320), (32, 128), 128),
           Inception(384, (192, 384), (48, 128), 128),
           nn.GlobalAvgPool2D(),nn.Dense(10))
    return(net)


X = nd.random.uniform(shape=(1, 1, 96, 96))
googlenet = GoogLeNet()
googlenet.initialize()
out = googlenet(X)
googlenet.summary(mx.nd.random.uniform(shape=(1, 1, 244, 244)))
```

```
--------------------------------------------------------------------------------
        Layer (type)                                Output Shape         Param #
================================================================================
               Input                            (1, 1, 244, 244)               0
        Activation-1                           (1, 64, 122, 122)               0
            Conv2D-2                           (1, 64, 122, 122)            3200
         MaxPool2D-3                             (1, 64, 61, 61)               0
        Activation-4                             (1, 64, 61, 61)               0
            Conv2D-5                             (1, 64, 61, 61)            4160
        Activation-6                            (1, 192, 61, 61)               0
            Conv2D-7                            (1, 192, 61, 61)          110784
         MaxPool2D-8                            (1, 192, 31, 31)               0
```

```
   Activation-9                  (1, 64, 31, 31)               0
      Conv2D-10                  (1, 64, 31, 31)           12352
  Activation-11                  (1, 96, 31, 31)               0
      Conv2D-12                  (1, 96, 31, 31)           18528
  Activation-13                 (1, 128, 31, 31)               0
      Conv2D-14                 (1, 128, 31, 31)          110720
  Activation-15                  (1, 16, 31, 31)               0
      Conv2D-16                  (1, 16, 31, 31)            3088
  Activation-17                  (1, 32, 31, 31)               0
      Conv2D-18                  (1, 32, 31, 31)           12832
   MaxPool2D-19                 (1, 192, 31, 31)               0
  Activation-20                  (1, 32, 31, 31)               0
      Conv2D-21                  (1, 32, 31, 31)            6176
   Inception-22                 (1, 256, 31, 31)               0
  Activation-23                 (1, 128, 31, 31)               0
      Conv2D-24                 (1, 128, 31, 31)           32896
  Activation-25                 (1, 128, 31, 31)               0
      Conv2D-26                 (1, 128, 31, 31)           32896
  Activation-27                 (1, 192, 31, 31)               0
      Conv2D-28                 (1, 192, 31, 31)          221376
  Activation-29                  (1, 32, 31, 31)               0
      Conv2D-30                  (1, 32, 31, 31)            8224
  Activation-31                  (1, 96, 31, 31)               0
      Conv2D-32                  (1, 96, 31, 31)           76896
   MaxPool2D-33                 (1, 256, 31, 31)               0
  Activation-34                  (1, 64, 31, 31)               0
      Conv2D-35                  (1, 64, 31, 31)           16448
   Inception-36                 (1, 480, 31, 31)               0
   MaxPool2D-37                 (1, 480, 16, 16)               0
  Activation-38                 (1, 192, 16, 16)               0
      Conv2D-39                 (1, 192, 16, 16)           92352
  Activation-40                  (1, 96, 16, 16)               0
      Conv2D-41                  (1, 96, 16, 16)           46176
  Activation-42                 (1, 208, 16, 16)               0
      Conv2D-43                 (1, 208, 16, 16)          179920
  Activation-44                  (1, 16, 16, 16)               0
      Conv2D-45                  (1, 16, 16, 16)            7696
  Activation-46                  (1, 48, 16, 16)               0
      Conv2D-47                  (1, 48, 16, 16)           19248
   MaxPool2D-48                 (1, 480, 16, 16)               0
  Activation-49                  (1, 64, 16, 16)               0
      Conv2D-50                  (1, 64, 16, 16)           30784
   Inception-51                 (1, 512, 16, 16)               0
  Activation-52                 (1, 160, 16, 16)               0
      Conv2D-53                 (1, 160, 16, 16)           82080
  Activation-54                 (1, 112, 16, 16)               0
      Conv2D-55                 (1, 112, 16, 16)           57456
  Activation-56                 (1, 224, 16, 16)               0
```

```
      Conv2D-57                            (1, 224, 16, 16)           226016
  Activation-58                             (1, 24, 16, 16)                0
      Conv2D-59                             (1, 24, 16, 16)            12312
  Activation-60                             (1, 64, 16, 16)                0
      Conv2D-61                             (1, 64, 16, 16)            38464
   MaxPool2D-62                            (1, 512, 16, 16)                0
  Activation-63                             (1, 64, 16, 16)                0
      Conv2D-64                             (1, 64, 16, 16)            32832
   Inception-65                            (1, 512, 16, 16)                0
  Activation-66                            (1, 128, 16, 16)                0
      Conv2D-67                            (1, 128, 16, 16)            65664
  Activation-68                            (1, 128, 16, 16)                0
      Conv2D-69                            (1, 128, 16, 16)            65664
  Activation-70                            (1, 256, 16, 16)                0
      Conv2D-71                            (1, 256, 16, 16)           295168
  Activation-72                             (1, 24, 16, 16)                0
      Conv2D-73                             (1, 24, 16, 16)            12312
  Activation-74                             (1, 64, 16, 16)                0
      Conv2D-75                             (1, 64, 16, 16)            38464
   MaxPool2D-76                            (1, 512, 16, 16)                0
  Activation-77                             (1, 64, 16, 16)                0
      Conv2D-78                             (1, 64, 16, 16)            32832
   Inception-79                            (1, 512, 16, 16)                0
  Activation-80                            (1, 112, 16, 16)                0
      Conv2D-81                            (1, 112, 16, 16)            57456
  Activation-82                            (1, 144, 16, 16)                0
      Conv2D-83                            (1, 144, 16, 16)            73872
  Activation-84                            (1, 288, 16, 16)                0
      Conv2D-85                            (1, 288, 16, 16)           373536
  Activation-86                             (1, 32, 16, 16)                0
      Conv2D-87                             (1, 32, 16, 16)            16416
  Activation-88                             (1, 64, 16, 16)                0
      Conv2D-89                             (1, 64, 16, 16)            51264
   MaxPool2D-90                            (1, 512, 16, 16)                0
  Activation-91                             (1, 64, 16, 16)                0
      Conv2D-92                             (1, 64, 16, 16)            32832
   Inception-93                            (1, 528, 16, 16)                0
  Activation-94                            (1, 256, 16, 16)                0
      Conv2D-95                            (1, 256, 16, 16)           135424
  Activation-96                            (1, 160, 16, 16)                0
      Conv2D-97                            (1, 160, 16, 16)            84640
  Activation-98                            (1, 320, 16, 16)                0
      Conv2D-99                            (1, 320, 16, 16)           461120
 Activation-100                             (1, 32, 16, 16)                0
     Conv2D-101                             (1, 32, 16, 16)            16928
 Activation-102                            (1, 128, 16, 16)                0
     Conv2D-103                            (1, 128, 16, 16)           102528
  MaxPool2D-104                            (1, 528, 16, 16)                0
```

```
      Activation-105                              (1, 128, 16, 16)               0
          Conv2D-106                              (1, 128, 16, 16)           67712
       Inception-107                              (1, 832, 16, 16)               0
       MaxPool2D-108                                (1, 832, 8, 8)               0
      Activation-109                                (1, 256, 8, 8)               0
          Conv2D-110                                (1, 256, 8, 8)          213248
      Activation-111                                (1, 160, 8, 8)               0
          Conv2D-112                                (1, 160, 8, 8)          133280
      Activation-113                                (1, 320, 8, 8)               0
          Conv2D-114                                (1, 320, 8, 8)          461120
      Activation-115                                 (1, 32, 8, 8)               0
          Conv2D-116                                 (1, 32, 8, 8)           26656
      Activation-117                                (1, 128, 8, 8)               0
          Conv2D-118                                (1, 128, 8, 8)          102528
       MaxPool2D-119                                (1, 832, 8, 8)               0
      Activation-120                                (1, 128, 8, 8)               0
          Conv2D-121                                (1, 128, 8, 8)          106624
       Inception-122                                (1, 832, 8, 8)               0
      Activation-123                                (1, 384, 8, 8)               0
          Conv2D-124                                (1, 384, 8, 8)          319872
      Activation-125                                (1, 192, 8, 8)               0
          Conv2D-126                                (1, 192, 8, 8)          159936
      Activation-127                                (1, 384, 8, 8)               0
          Conv2D-128                                (1, 384, 8, 8)          663936
      Activation-129                                 (1, 48, 8, 8)               0
          Conv2D-130                                 (1, 48, 8, 8)           39984
      Activation-131                                (1, 128, 8, 8)               0
          Conv2D-132                                (1, 128, 8, 8)          153728
       MaxPool2D-133                                (1, 832, 8, 8)               0
      Activation-134                                (1, 128, 8, 8)               0
          Conv2D-135                                (1, 128, 8, 8)          106624
       Inception-136                               (1, 1024, 8, 8)               0
 GlobalAvgPool2D-137                               (1, 1024, 1, 1)               0
           Dense-138                                       (1, 10)           10250
================================================================================
Parameters in forward computation graph, duplicate included
   Total params: 5977530
   Trainable params: 5977530
   Non-trainable params: 0
Shared params in forward computation graph: 0
Unique parameters in model: 5977530
--------------------------------------------------------------------------------
```

2. **What is the minimum image size for GoogLeNet to work?**

   96

3. **Compare the model parameter sizes of AlexNet, VGG, and NiN with GoogLeNet. How**

do the latter two network architectures significantly reduce the model parameter size?

- GoogLeNet has 5977530 ~ 6M params;
- NiN has ~ 2M paras;
- VGG has ~ 129M paras;
- AlexNet has ~46M params;

Dense layer in AlexNet and VGG are the one which needs the most parameters. While NiN and GoogLeNet use `GlobalAvgPool2D` to pool the weights to one dimension first, then apply a `Dense` layer.

4. **Why do we need a large range convolution initially?**