Dynamic Resolution Network (Supplementary Material)

Mingjian Zhu\textsuperscript{1,2,4,5,}, Kai Han\textsuperscript{2,3,}, Enhua Wu\textsuperscript{3,6}, Qiu Lin Zhang\textsuperscript{7}, Ying Nie\textsuperscript{2}, Zhenzhong Lan\textsuperscript{4,5}, Yunhe Wang\textsuperscript{2}\textsuperscript{†}
\textsuperscript{1}\textsuperscript{Zhejiang University.} \textsuperscript{2}\textsuperscript{Huawei Noah's Ark Lab.}\textsuperscript{3}\textsuperscript{State Key Lab of Computer Science, ISCAS & University of Chinese Academy of Sciences.}\textsuperscript{4}\textsuperscript{School of Engineering, Westlake University.}\textsuperscript{5}\textsuperscript{Institute of Advanced Technology, Westlake Institute for Advanced Study.}\textsuperscript{6}\textsuperscript{University of Macau.}\textsuperscript{7}\textsuperscript{BUPT.}
zhumingjian@zju.edu.cn, \{hankai,weh\}@ios.ac.cn, lanzhenzhong@westlake.edu.cn, yunhe.wang@huawei.com

1 Details of predictor architectures

We utilize the basic block of resnet to construct the predictor, i.e., 4 basic blocks are stacked to form the predictor network in our paper. Here we build 4 predictor architectures with fewer parameters and FLOPs and we compare their performances as follows. We can see that the predictor with fewer flops leads to slight accuracy degradation.

(1) Predictor-Architecture-1: The original predictor in our paper. The parameters of the first convolution are Conv2d (3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False).

(2) Predictor-Architecture-2: We reduce the blocks of the predictor in (1) to construct a new predictor. We retain only 2 blocks.

(3) Predictor-Architecture-3: We increase the stride of the first convolution in (2) to 4. Thus, the parameters of the first convolution are Conv2d (3, 64, kernel_size=(7, 7), stride=(4, 4), padding=(3, 3), bias=False).

(4) Predictor-Architecture-4: We construct the predictor with only two convolutions.

<table>
<thead>
<tr>
<th>Predictor</th>
<th>Predictor FLOPs</th>
<th>Total FLOPs</th>
<th>Acc</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>0.29G</td>
<td>3.35G</td>
<td>82.5%</td>
</tr>
<tr>
<td>(2)</td>
<td>0.17G</td>
<td>3.23G</td>
<td>82.0%</td>
</tr>
<tr>
<td>(3)</td>
<td>0.04G</td>
<td>3.10G</td>
<td>82.0%</td>
</tr>
<tr>
<td>(4)</td>
<td>0.09G</td>
<td>3.15G</td>
<td>81.4%</td>
</tr>
</tbody>
</table>

Table 1: comparison of different Predictors.

The details of these predictors are shown as follows:

(1) Predictor-Architecture-1

\begin{verbatim}
ResNet(
  (conv1)
  (bn1)
\end{verbatim}

\textsuperscript{*}Equal contribution. \textsuperscript{†}Corresponding author. This research has been supported by the Key R&D program of Zhejiang Province (Grant No. 2021C03139). This work was supported by NSFC (62072449, 61632003), Guangdong-Hongkong-Macao Joint Research Grant (2020B151530004), and Macao FDCT (0018/2019/ AKP).

(2) Predictor-Architecture-2

ResNet(
    (conv1)
    (bn1)
    (relu)
    (maxpool)
    (layer1): Sequential(
        (0): BasicBlock(
            (conv1)
            (bn1)
            (relu)
            (conv2)
            (bn2)
        )
    )
    (layer2): Sequential(
        (0): BasicBlock(
            (conv1)
            (bn1)
            (relu)
            (conv2)
            (bn2)
            (downsample): Sequential(
                (0): Conv2d()
                (1): BatchNorm2d()
            )
        )
    )
    (layer3): Sequential(
        (0): BasicBlock(
            (conv1)
            (bn1)
            (relu)
            (conv2)
            (bn2)
            (downsample): Sequential(
                (0): Conv2d()
                (1): BatchNorm2d()
            )
        )
    )
    (layer4): Sequential(
        (0): BasicBlock(
            (conv1)
            (bn1)
            (relu)
            (conv2)
            (bn2)
            (downsample): Sequential(
                (0): Conv2d()
                (1): BatchNorm2d()
            )
        )
    )
    (avgpool): AdaptiveAvgPool2d()
    (dropout): Dropout()
    (fc): Linear()
)
(0): BasicBlock(
  (conv1)
  (bn1)
  (relu)
  (conv2)
  (bn2)
)
(layer2): Sequential(
  (0): BasicBlock(
    (conv1)
    (bn1)
    (relu)
    (conv2)
    (bn2)
    (downsample): Sequential(
      (0): Conv2d()
      (1): BatchNorm2d()
    )
  )
)(avgpool)
(dropout)
(fc)
)

(3) Predictor-Architecture-3

ResNet(
  (conv1)
  (bn1)
  (relu)
  (maxpool)
  (layer1): Sequential(
    (0): BasicBlock(
      (conv1)
      (bn1)
      (relu)
      (conv2)
      (bn2)
    )
  )
  (layer2): Sequential(
    (0): BasicBlock(
      (conv1)
      (bn1)
      (relu)
      (conv2)
      (bn2)
      (downsample): Sequential(
        (0): Conv2d()
        (1): BatchNorm2d()
      )
    )
  )
)(avgpool)
(dropout)
(fc)
)

(4) Predictor-Architecture-4
ResNet
(conv1)
(bn1)
(relu)
(maxpool)
(conv2)
(bn2)
(relu)
(avgpool)
(dropout)
(fc)

2 ImageNet-100 Categories.

```
ap1494475 n02095314 n02108915 n02177972 n02787622 n02971356 n03482405 n03899768 n04204238 n04509417
a01644900 n02097047 n02111889 n02219486 n02791124 n03065424 n03496892 n03908714 n04235860 n04542943
a01768244 n02097298 n02113023 n02229544 n02793495 n03124043 n03527444 n03977966 n04251144 n04548280
a01770393 n02099712 n02113799 n02443114 n02814860 n03180011 n03544143 n03982430 n04258138 n04554684
a01775062 n02101556 n02113978 n02494079 n02815834 n03201208 n03584829 n04041544 n04265275 n04590129
a01797886 n02102177 n02115913 n02504013 n02825657 n03216848 n03598930 n04049303 n04270147 n07717410
a01847000 n02105412 n02124075 n02606052 n02835271 n03372029 n03617480 n04081281 n04275548 n07920052
a01910747 n02106550 n02128757 n02667093 n02837789 n03461385 n03637318 n04116512 n04417672 n09193705
a01978287 n02108000 n02129604 n02687172 n02965783 n03476684 n03710721 n04118538 n04458633 n10565667
a02085782 n02108422 n02165105 n02699494 n02966687 n03476991 n03804744 n04120489 n04487081 n12144580
```