

9/12 給耐能的報錯

tags: 耐能加速棒

目的

為了我們的作品，我們找了一個pytorch模型(<https://github.com/tiangexiang/CurveNet>)來使用，這是用來預測點雲的模型。預計先在colab訓練，得到一個.pth檔後，再將其轉成ONNX檔以利後續用在KL-520加速棒上。但在轉成ONNX時就遇到了滿多錯誤的，以下是主要的內容。如果有什麼描述的不清楚的請跟我說，謝謝。

過程

1. 訓練模型

1. 下載這個pytorch模型的repo(<https://github.com/tiangexiang/CurveNet>)
2. 將其上傳到Google Drive
3. 開啟colab，掛載到Google Drive資料夾

```
1 from google.colab import drive
2 drive.mount('/content/drive')
```

4. 執行 main_cls.py 來訓練模型

```
1 import os
2 os.chdir("/content/drive/MyDrive/CompetitionMMwave/CurveNet-main/core")
3 !python main_cls.py --exp_name=0909_1 --epochs=1
```

5. 會得到以下幾個模型，我們選擇 model.pth來轉換成ONNX檔，將其下載。



2. 將model.pth轉成ONNX

1. 準備一個資料夾，裡面附有以下檔案

名稱	修改日期	類型
__pycache__	2021/9/9 下午 09:48	檔案資料夾
models	2021/9/3 下午 09:40	檔案資料夾
curvenet_util.py	2021/9/9 下午 09:48	Python File
model.pth	2021/9/9 下午 04:11	PTH 檔案
pytorch2onnx.py	2021/9/9 下午 10:50	Python File
walk.py	2021/9/9 下午 09:36	Python File

model.pth 是訓練好的模型

curvenet_util.py 、 walk.py 以及 models資料夾 是pytorch模型的repo
(<https://github.com/tiangexiang/CurveNet>)裡的檔案

pytorch2onnx.py 是自己創的py檔，內容在下面的Appendix
(<https://hackmd.io/MUp1gh1kRy2vr0oBtQvwuw#Appendix>).

2. 在python=3.8環境下執行 pytorch2onnx.py

3. 有錯誤，如下：

```

1 | Traceback (most recent call last):
2 |   File "pytorchTraining.py", line 110, in <module>
3 |     model.load_state_dict(torch.load(path))
4 |   File "/usr/local/lib/python3.7/dist-packages/torch/nn/modules/module.py", line
5 |     self.__class__.__name__, "\n\t".join(error_msgs)))
6 | RuntimeError: Error(s) in loading state_dict for CurveNet:
7 |   Missing key(s) in state_dict: "lpfa.mlp.0.0.weight", "lpfa.mlp.0.1.weight",
8 |   Unexpected key(s) in state_dict: "module.lpfa.mlp.0.0.weight", "module.lpfa.

```

類似這裡的錯誤 (<https://discuss.pytorch.org/t/solved-keyerror-unexpected-key-module-encoder-embedding-weight-in-state-dict/1686/17>)，底下的有個回答如下：



Alon



Wyn_Mew

Jul '18

Instead of deleting the "module." string from all the state_dict keys, you can save your model with:

```
torch.save(model.module.state_dict(), path_to_file)
```

instead of

```
torch.save(model.state_dict(), path_to_file)
```

that way you don't get the "module." string to begin with...

2 Replies ^

11 ❤️



4. 依照回答，要重新save model，所以回到colab。

5. 把 main_cls.py 中的第145行，如下：

```
1 | torch.save(model.state_dict(), '../checkpoints/%s/models/model.pth' % args.exp_n
```

改成如下：

```
1 | torch.save(model.module.state_dict(), '../checkpoints/%s/models/model.pth' % arg
```

3. 在colab中執行下列指令，訓練模型並儲存：

```

1 | import os
2 | os.chdir("/content/drive/MyDrive/CompetitionMMwave/CurveNet-main/core")
3 | !python main_cls.py --exp_name=0909_1 --epochs=1

```

4. 一樣在google drive中， /CurveNet-main/checkpoint/ 可以找到 model.pth，並下載。

5. 把 model.pth 換掉後，執行 pytorch2onnx.py，結果有錯誤如下：

發生例外狀況: IndexError

Dimension out of range (expected to be in range of [-2, 1], but got 2)

```
File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 468,
    start_index = start_index.squeeze().unsqueeze(2)
File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 368,
    curves = self.curvegrouping(x, xyz, idx[:, :, 1:]) # avoid self-loop
File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line
    l1_xyz, l1_points = self.cic11(xyz, l0_points)
File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line
    torch.onnx.export(torch_model,                # model being run
```

```
464         _, start_index = torch.topk(x_att,
465                                     self.curve_num,
466                                     dim=2,
467                                     sorted=False)
468         start_index = start_index.squeeze().unsqueeze(2)
```

發生例外狀況: IndexError ×

Dimension out of range (expected to be in range of [-2, 1], but got 2)

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 468, in forward
start_index = start_index.squeeze().unsqueeze(2)

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 368, in forward
curves = self.curvegrouping(x, xyz, idx[:, :, 1:]) # avoid self-loop

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 47, in forward
l1_xyz, l1_points = self.cic11(xyz, l0_points)

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 99, in <module>
torch.onnx.export(torch_model, # model being run

6. 接下來就嘗試修該code看看。把 curvenet_util.py 的第468行（如上圖），unsqueeze(2) 改成 unsqueeze(1)，再執行。換 wak1.py 有錯，如下：

```
87         flatten_x = x.view(bn * tot_points, -1)
88         batch_offset = torch.arange(0, bn, device=torch.device('cuda')).detach() * tot_points
```

發生例外狀況: RuntimeError ×

Found no NVIDIA driver on your system. Please check that you have an NVIDIA GPU and installed a driver from <http://www.nvidia.com/Download/index.aspx>

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\wak1.py", line 88, in forward
batch_offset = torch.arange(0, bn, device=torch.device('cuda')).detach() * tot_points

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 470, in forward
curves = self.wak(xyz, x, idx, start_index) #bs, c, c_n, c_l

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 368, in forward
curves = self.curvegrouping(x, xyz, idx[:, :, 1:]) # avoid self-loop

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 47, in forward
l1_xyz, l1_points = self.cic11(xyz, l0_points)

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 99, in <module>
torch.onnx.export(torch_model, # model being run

7. 把 cuda 改成 cpu，再執行一次 pytorch2onnx.py。一樣在 wak1.py 有錯，如下：

```

106         else:
107             # dynamic momentum
108             cat_feature = torch.cat((cur_feature.squeeze(), pre_feature.squeeze()),dim=1)
109             att_feature = F.softmax(self.momentum_mlp(cat_feature),dim=1).view(bn, 1, self.curve_num, 2) # bs, 1, n, 2

```

發生例外狀況: RuntimeError ×

Expected 3-dimensional input for 3-dimensional weight [2, 32, 1], but got 2-dimensional input of size [16, 200] instead

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\walk.py", line 109, in forward
 att_feature = F.softmax(self.momentum_mlp(cat_feature),dim=1).view(bn, 1, self.curve_num, 2) # bs, 1, n, 2

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 470, in forward
 curves = self.walk(xyz, x, idx, start_index) #bs, c, c_n, c_l

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py", line 368, in forward
 curves = self.curvegrouping(x, xyz, idx[:, :, 1:]) # avoid self-loop

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 47, in forward
 l1_xyz, l1_points = self.cic11(xyz, l0_points)

File "C:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line 99, in <module>
 torch.onnx.export(torch_model, # model being run

8. 後來發現把 pytorch2onnx.py 中的

```
1 | x = torch.randn(1, 3, 1024, requires_grad=True)
```

改成如下，可以跑出更多東西，所以先把前面改的 `unsqueeze(1)` 改回 `unsqueeze(2)`

```
1 | x = torch.randn(32, 3, 1024, requires_grad=True)
```

執行後如下，有錯誤：

```

C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\cuda\__init__.py:52: Us
  return torch._C._cuda_getDeviceCount() > 0
x: tensor([[-1.0610, -0.5267,  1.4753, ..., -0.4559, -0.1569, -1.5430],
          [ 0.0778, -0.4320, -0.5244, ...,  0.4428, -0.1741,  0.3060],
          [ 0.1440, -1.5503,  1.1274, ..., -1.0515,  0.0273, -1.2841]],

          [[-0.7762,  0.2611, -1.1352, ..., -1.3489,  1.0865,  0.4396],
          [-0.4481,  2.0752,  0.4898, ...,  0.0846, -0.0680,  0.0815],
          [ 0.6015,  2.7668,  0.3792, ...,  1.6718, -0.2551,  2.4854]],

          [[-1.4961,  0.6301,  0.2529, ..., -1.5639,  0.3833, -0.6893],
          [-0.4424, -0.7423,  0.6153, ..., -1.9716,  0.1808, -0.8182],
          [ 0.5502, -0.1931,  0.7892, ..., -0.0410, -1.2228,  0.1584]],

          ...,

          [[-0.9705, -1.4239, -0.4263, ...,  0.1071,  0.0304, -1.5994],
          [-1.3054, -0.8234, -0.7768, ...,  0.6917, -0.3518, -0.1506],
          [-1.1786, -0.4557, -0.1489, ..., -1.3476,  2.7490, -0.3241]],

          [[ 0.5510, -1.8050,  1.0268, ..., -1.0423, -0.6780, -1.5962],
          [ 0.3624,  0.1122,  0.1071, ...,  1.5958,  0.6209, -1.3937],
          [ 1.4254,  0.1170,  0.1670, ..., -1.2565, -0.9526, -2.0219]],

          [[ 2.6874,  1.1777, -1.6811, ..., -0.2441, -0.0778,  1.7057],
          [ 0.0602,  0.6392, -0.3591, ...,  0.7949,  0.6193,  0.9046],
          [ 0.3005,  1.2581,  0.4919, ...,  0.4723, -1.3764,  0.5513]]],
  requires_grad=True)
torch.Size([32, 3, 1024])
torch.Size([32, 16, 1024])
torch.Size([32, 16, 1024])
torch.Size([32, 32, 1024])
torch.Size([32, 32, 1024])
torch.Size([32, 64, 256])
torch.Size([32, 64, 256])
torch.Size([32, 128, 64])
torch.Size([32, 128, 64])
tensor([ [ 8.0885, -4.3205, -0.3741, ..., -2.1033, -0.7325, -2.5036],
        [10.6468, -3.3417,  2.4449, ..., -4.2334,  0.5905, -3.7904],
        [14.5540, -5.7213,  3.9037, ..., -2.1741,  2.3348, -2.4426],
        ...,
        [ 9.1924, -3.6008,  3.4110, ..., -2.6812,  0.3097, -3.6180],
        [11.4883, -5.2502,  0.2312, ..., -2.5288, -0.6245, -2.4642],
        [ 9.1515, -4.6202,  1.6596, ..., -1.7074, -0.9047, -3.9957]],
  grad_fn=<AddmmBackward>)
c:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py:213: TracerWarning:
  print(x.size())
torch.Size([32, 3, 1024])
c:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py:356: TracerWarning:
We can't record the data flow of Python values, so this value will be treated as a con
  if xyz.size(-1) != self.npoint:
torch.Size([32, 16, 1024])
torch.Size([32, 16, 1024])
torch.Size([32, 32, 1024])

```

```

torch.Size([32, 32, 1024])
c:\Users\chris\CompetitionMMwave\pytorchTraining\curvenet_util.py:90: TracerWarning: C
can't record the data flow of Python values, so this value will be treated as a consta
    farthest = torch.randint(0, N, (B,), dtype=torch.long).to(device) * 0
torch.Size([32, 64, 256])
torch.Size([32, 64, 256])
torch.Size([32, 128, 64])
torch.Size([32, 128, 64])
C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_opset9.py
will produce incorrect results.
    warnings.warn("Exporting aten::index operator of advanced indexing in opset " +
C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_opset9.py
    warnings.warn("Exporting aten::index operator with indices of type Byte. ")
C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_opset9.py
    warnings.warn("This model contains a squeeze operation on dimension " + str(squeeze_
Traceback (most recent call last):
  File "C:\Users\chris\anaconda3\envs\python38\lib\runpy.py", line 192, in _run_module
    return _run_code(code, main_globals, None,
  File "C:\Users\chris\anaconda3\envs\python38\lib\runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "c:\Users\chris\.vscode\extensions\ms-python.python-2021.9.1191016588\pythonFil
    cli.main()
  File "c:\Users\chris\.vscode\extensions\ms-python.python-2021.9.1191016588\pythonFil
    run()
  File "c:\Users\chris\.vscode\extensions\ms-python.python-2021.9.1191016588\pythonFil
    runpy.run_path(target_as_str, run_name=compat.force_str("__main__"))
  File "C:\Users\chris\anaconda3\envs\python38\lib\runpy.py", line 262, in run_path
    return _run_module_code(code, init_globals, run_name,
  File "C:\Users\chris\anaconda3\envs\python38\lib\runpy.py", line 95, in _run_module_
    _run_code(code, mod_globals, init_globals,
  File "C:\Users\chris\anaconda3\envs\python38\lib\runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "c:\Users\chris\CompetitionMMwave\pytorchTraining\pytorchTrainingTest.py", line
    torch.onnx.export(torch_model,                # model being run
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\__init__.p
    return utils.export(model, args, f, export_params, verbose, training,
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\utils.py",
    _export(model, args, f, export_params, verbose, training, input_names, output_name
    graph = _optimize_graph(graph, operator_export_type,
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\utils.py",
    graph = torch._C._jit_pass_onnx(graph, operator_export_type)
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\__init__.p
    return utils._run_symbolic_function(*args, **kwargs)
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\utils.py",
    return symbolic_fn(g, *inputs, **attrs)
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_h
    args = [_parse_arg(arg, arg_desc) for arg, arg_desc in zip(args, arg_descriptors)]
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_h
    args = [_parse_arg(arg, arg_desc) for arg, arg_desc in zip(args, arg_descriptors)]
  File "C:\Users\chris\anaconda3\envs\python38\lib\site-packages\torch\onnx\symbolic_h
    raise RuntimeError("Failed to export an ONNX attribute '" + v.node().kind() +
RuntimeError: Failed to export an ONNX attribute 'onnx::Gather', since it's not consta

```

目前到這裡，還沒有找出方法。

附註：

也有將這個 `pytorch2onnx.py` 放到colab上執行看看（用GPU），可是嘗試了很多方法，還是會一直跑出以下這個錯誤：

```
RuntimeError: Expected all tensors to be on the same device, but found at least two de
```



另外，我看一般給 `torch.onnx.export()` 的輸入都是像這樣有4個參數，分別是（`batch_size`, `channels`, `hight`, `width`），就像圖片：

```
1 | x = torch.randn(1, 3, 224, 224, requires_grad=True)
```

但點雲的輸入似乎比較特別，跟圖片不一樣。只有三個，分別是（`batch_size`, `num_dims`, `num_points`）。所以會給這樣輸入：

```
1 | x = torch.randn(32, 3, 1024, requires_grad=True)
```

因為剛接觸沒多久，所以對ONNX還有pytorch不是到非常了解，只大概知道怎麼轉換，但不知道有那些限制。

謝謝幫忙~~

Appendix

`pytorch2onnx.py` (<http://pytorch2onnx.py>).

（這是依照這裡 (<https://hackmd.io/MUp1gh1kRy2vr0oBtQvwuw#Appendix>)來建立的）

（`class CurveNet(nn.Module)` 是複製這裡的內容

(https://github.com/tiangexiang/CurveNet/blob/main/core/models/curvenet_cls.py).


```

1  import torch
2  import torch.nn as nn
3  import torchvision
4  import torch.nn.functional as F
5  from curvenet_util import *
6
7  # Define a convolution neural network
8  curve_config = {
9      'default': [[100, 5], [100, 5], None, None],
10     'long': [[10, 30], None, None, None]
11 }
12
13 class CurveNet(nn.Module):
14     def __init__(self, num_classes=40, k=20, setting='default'):
15         super(CurveNet, self).__init__()
16
17         assert setting in curve_config
18
19         additional_channel = 32
20         self.lpfa = LPFA(9, additional_channel, k=k, mlp_num=1, initial=True)
21
22         # encoder
23         self.cic11 = CIC(npoint=1024, radius=0.05, k=k, in_channels=additional_channel, output_channels=1024)
24         self.cic12 = CIC(npoint=1024, radius=0.05, k=k, in_channels=64, output_channels=1024)
25
26         self.cic21 = CIC(npoint=1024, radius=0.05, k=k, in_channels=64, output_channels=1024)
27         self.cic22 = CIC(npoint=1024, radius=0.1, k=k, in_channels=128, output_channels=1024)
28
29         self.cic31 = CIC(npoint=256, radius=0.1, k=k, in_channels=128, output_channels=256)
30         self.cic32 = CIC(npoint=256, radius=0.2, k=k, in_channels=256, output_channels=256)
31
32         self.cic41 = CIC(npoint=64, radius=0.2, k=k, in_channels=256, output_channels=64)
33         self.cic42 = CIC(npoint=64, radius=0.4, k=k, in_channels=512, output_channels=64)
34
35         self.conv0 = nn.Sequential(
36             nn.Conv1d(512, 1024, kernel_size=1, bias=False),
37             nn.BatchNorm1d(1024),
38             nn.ReLU(inplace=True))
39         self.conv1 = nn.Linear(1024 * 2, 512, bias=False)
40         self.conv2 = nn.Linear(512, num_classes)
41         self.bn1 = nn.BatchNorm1d(512)
42         self.dp1 = nn.Dropout(p=0.5)
43
44     def forward(self, xyz):
45         l0_points = self.lpfa(xyz, xyz)
46
47         l1_xyz, l1_points = self.cic11(xyz, l0_points)
48         l1_xyz, l1_points = self.cic12(l1_xyz, l1_points)
49
50         l2_xyz, l2_points = self.cic21(l1_xyz, l1_points)
51         l2_xyz, l2_points = self.cic22(l2_xyz, l2_points)
52
53         l3_xyz, l3_points = self.cic31(l2_xyz, l2_points)
54         l3_xyz, l3_points = self.cic32(l3_xyz, l3_points)

```

```

55
56     l4_xyz, l4_points = self.cic41(l3_xyz, l3_points)
57     l4_xyz, l4_points = self.cic42(l4_xyz, l4_points)
58
59     x = self.conv0(l4_points)
60     x_max = F.adaptive_max_pool1d(x, 1)
61     x_avg = F.adaptive_avg_pool1d(x, 1)
62
63     x = torch.cat((x_max, x_avg), dim=1).squeeze(-1)
64     x = F.relu(self.bn1(self.conv1(x).unsqueeze(-1)), inplace=True).squeeze(-1)
65     x = self.dp1(x)
66     x = self.conv2(x)
67     return x
68
69
70 # Create the CurveNet model by using the above model definition.
71 torch_model = CurveNet().cpu()
72 #print(torch_model)
73
74 model_path = 'C:/Users/chris/CompetitionMMwave/pytorchTraining/model.pth'
75 batch_size = 1    # just a random number
76
77
78 # Initialize model with the pretrained weights
79 map_location = lambda storage, loc: storage
80 if torch.cuda.is_available():
81     map_location = None
82 torch_model.load_state_dict(torch.load(model_path, map_location=map_location))
83
84 # set the model to inference mode
85 torch_model.eval()
86 #print(torch_model)
87 #print("-----")
88
89 # Input to the model
90 #x = torch.randn(batch_size, 3, 1024, requires_grad=True)
91 x = torch.randn(32, 3, 1024, requires_grad=True)
92
93 print('x:',x)
94
95 #torch_out = torch_model(x)
96 #print(torch_out)
97
98 import torch.onnx
99
100 # Export the model
101 torch.onnx.export(torch_model,                # model being run
102                  x,                          # model input (or a tuple for multiple inputs)
103                  "model.onnx",               # where to save the model (can be a file or folder)
104                  export_params=True,          # store the trained parameter weights inside the model file
105                  opset_version=11,           # the ONNX version to export the model to
106                  do_constant_folding=True,    # whether to execute constant folding for optimization
107                  input_names = ['input'],     # the model's input names
108                  output_names = ['output'],   # the model's output names
109                  dynamic_axes={'input' : {0 : 'batch_size'}, # variable length axes

```

110
111
112

