ConvTranspose2d 포팅
Python ver.

염지현

1. ConvTranspose2d 설계

* ConvTranspose2d

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

1) Output size 연산
2) Padding을 추가한 input 만들기
    1) 모든 padding이 추가된 사이즈 연산
3) Kernel 상하좌우 반전 시키기
4) Convolution 연산 진행하기

1. ConvTranspose2d 설계

1) weight, bias 불러오기

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

```python
""" kernel 불러오기 """
import os
kernel_dir = ('./parameter/kernel/last')
kernel = torch.zeros([56, 1, 9, 9]).numpy()
for i in range(56):
    kernel_file = kernel_dir + '/kernel' + str(i) +'_0.txt'
    kernel_ = np.loadtxt(kernel_file)
    kernel[i][0] = kernel_

kernel_T = torch.zeros([56, 1,9,9]).numpy()
for index in range(56):
    for i in range(9):
        for j in range(9):
            kernel_T[index][0][i][j] = kernel[index][0][9-1-i][9-1-j]

""" bias 불러오기 """
bias = torch.zeros([1]).numpy()
bias_ = np.loadtxt('./parameter/bias/last/bias0_0.txt')
bias = bias_
bias = np.expand_dims(bias, axis = 1)
_bias = torch.zeros([1])
_bias[0] = bias[0]
```

1. ConvTranspose2d 설계

2) Padding 추가

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

```python
""" padding 추가 """
scale = 2

w = 1920
h = 1080
kernel_size = 9
pad = 4
_pad = 1 * (kernel_size - 1) - pad
stride = 2
input = mid.cpu().numpy()
output = Variable(torch.zeros(56,1,h*scale, w*scale)).numpy()

padding = torch.zeros(56, 1, 2 * h - 1 + (_pad * 2) + 1, stride * w - 1 + (_pad * 2) + 1).numpy()
print(padding.shape)
def padding_init(input, padding, in_ch, out_ch, kernel_size, stride, pad, w,h):
    _pad = 1 * (kernel_size - 1) - pad

    for index in range(in_ch):

        px = _pad
        py = _pad
        for i in range(h):
            for j in range(w):
                padding[index][0][py][px] = input[0][index][i][j]
                px = px + stride
            px = _pad
            py = py + stride

padding_init(input, padding, 56, 1, 9, 2, 4, w, h)
```

1. ConvTranspose2d 설계

3) Convolution 연산 진행

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

```python
""" convolution 연산 진행 """

for index in range(56):
    print('channel num:', index)
    for i in range(0, h * scale):
        h_start = i * 1
        h_end = h_start + kernel_size
        for j in range(0, w * scale):
            w_start = j * 1
            w_end = w_start + kernel_size
            output[index, 0, i, j] = np.sum(padding[index,0, h_start:h_end, w_start:w_end] * kernel_T[index, 0])
```

1. ConvTranspose2d 설계

4) 후처리

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

```python
""" 채널별 output 값 하나의 채널로 합치기 """
r = torch.zeros([1, 1, 2160, 3840]).numpy() # r: results
for i in range(56):
    r = output[i] + r
```

```python
""" bias 추가하기 """
a = r + bias
```

```python
""" tensor 변환하기 """
b =torch.tensor(a)
b.clamp(0.0, 1.0)
```

```
tensor([[[[0.2087, 0.2071, 0.2085,  ..., 0.2776, 0.2835, 0.2788],
          [0.2065, 0.2090, 0.2096,  ..., 0.2793, 0.2876, 0.2812],
          [0.2101, 0.2136, 0.2149,  ..., 0.2785, 0.2889, 0.2815],
          ...,
          [0.2476, 0.2504, 0.2534,  ..., 0.3890, 0.3890, 0.3846],
          [0.2569, 0.2587, 0.2627,  ..., 0.3868, 0.3877, 0.3819],
          [0.2682, 0.2650, 0.2679,  ..., 0.3759, 0.3781, 0.3784]]]],
       dtype=torch.float64)
```

2. 결과 비교

* torch.nn.ConvTraspose2d - jihyun's code 결과 비교

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

torch.nn.Convtranspose2d와 jihyun's python code 비교하기

```
In [72]:
1 torch_r_arr = torch_r.cpu().detach().numpy()
2 sub = np.around(torch_r_arr, 5) - np.around(a,5)

In [73]:
1 sub.max()

Out[73]: 1.0029792785637426e-05

In [74]:
1 sub.min()

Out[74]: -1.0029792785637426e-05
```

3. 난관 봉착

* preds - jihyun's code 결과 비교

```
(last_part): ConvTranspose2d(56, 1, kernel_size=(9, 9), stride=(2, 2), padding=(4, 4), output_padding=(1, 1))
```

preds와 jihyun's python code 비교

```
In [82]:  1  sub_2 = preds.cpu().numpy() - a

In [83]:  1  sub_2.max()
Out[83]: 0.4891911745071411

In [84]:  1  sub_2.min()
Out[84]: -0.39943212270736694
```

4. 피드백