

Operation

- View/reshape
- Squeeze/unsqueeze
- Transpose/t/permute
- Expand/repeat

View reshape

- Lost dim information

```
1 In [13]: a=torch.rand(4,1,28,28)
2
3 In [14]: a.shape
4 Out[14]: torch.Size([4, 1, 28, 28])
5
6 In [15]: a.view(4,28*28)
7 Out[15]:
8 tensor([[0.9020, 0.5015, 0.1010, ..., 0.4946, 0.1665, 0.5352],
9         [0.3386, 0.5002, 0.7858, ..., 0.1595, 0.9348, 0.3489],
10        [0.6014, 0.9255, 0.4829, ..., 0.4963, 0.1805, 0.7925],
11        [0.8898, 0.2387, 0.7560, ..., 0.1687, 0.3041, 0.3767]])
12
13 In [16]: a.view(4,28*28).shape
14 Out[16]: torch.Size([4, 784])
15
16 In [17]: a.view(4*28, 28).shape
17 Out[17]: torch.Size([112, 28])
18
19 In [18]: a.view(4*1, 28, 28).shape
20 Out[18]: torch.Size([4, 28, 28])
21
22 In [19]: b=a.view(4,784)
23
24 In [20]: b.view(4, 28, 28 ,1) # Logic Bug
```

Flexible but prone to corrupt



```
1 In [22]: a.view(4,783)
2 -----
3 RuntimeError                                Traceback (most recent call last)
4 <ipython-input-22-a68cdf9a2a6b> in <module>()
5 ----> 1 a.view(4,783)
6
7 RuntimeError: shape '[4, 783]' is invalid for input of size 3136
```

Squeeze v.s. unsqueeze



unsqueeze

`[-a.dim()-1, a.dim()+1)`

`[-5, 5)`

```
1 In [34]: a.shape
2 Out[34]: torch.Size([4, 1, 28, 28])
3
4 In [35]: a.unsqueeze(0).shape
5 Out[35]: torch.Size([1, 4, 1, 28, 28])
6
7 In [36]: a.unsqueeze(-1).shape
8 Out[36]: torch.Size([4, 1, 28, 28, 1])
9
10 In [37]: a.unsqueeze(4).shape
11 Out[37]: torch.Size([4, 1, 28, 28, 1])
12
13 In [38]: a.unsqueeze(-4).shape
14 Out[38]: torch.Size([4, 1, 1, 28, 28])
15
16 In [39]: a.unsqueeze(-5).shape
17 Out[39]: torch.Size([1, 4, 1, 28, 28])
18
19 In [40]: a.unsqueeze(5).shape
20 RuntimeError: Dimension out of range
21     (expected to be in range of [-5, 4], but got 5)
22
```

$[-5, 5)$



Pos. Idx	0	1	2	3
	4	3	28	28
Neg. Idx	-4	-3	-2	-1



```
1 In [46]: a=torch.tensor([1.2,2.3])
2
3 In [47]: a.unsqueeze(-1)
4 Out[47]:
5 tensor([[1.2000],
6         [2.3000]])
7
8 In [49]: a.unsqueeze(0)
9 Out[49]: tensor([[1.2000, 2.3000]])
10
```

For example



```
1 In [51]: b=torch.rand(32)
2
3 In [52]: f=torch.rand(4,32,14,14)
4
5 In [54]: b=b.unsqueeze(1).unsqueeze(2).unsqueeze(0)
6
7 In [55]: b.shape
8 Out[55]: torch.Size([1, 32, 1, 1])
9
```

squeeze

```
1 In [60]: b.shape
2 Out[60]: torch.Size([1, 32, 1, 1])
3
4 In [61]: b.squeeze().shape
5 Out[61]: torch.Size([32])
6
7 In [62]: b.squeeze(0).shape
8 Out[62]: torch.Size([32, 1, 1])
9
10 In [63]: b.squeeze(-1).shape
11 Out[63]: torch.Size([1, 32, 1])
12
13 In [64]: b.squeeze(1).shape
14 Out[64]: torch.Size([1, 32, 1, 1])
15
16 In [65]: b.squeeze(-4).shape
17 Out[65]: torch.Size([32, 1, 1])
18
```


Expand / repeat


- Expand: broadcasting
- Repeat: memory copied

Expand/expand_as

```
1 In [68]: a=torch.rand(4,32,14,14)
2
3 In [73]: b.shape
4 Out[73]: torch.Size([1, 32, 1, 1])
5
6 In [70]: b.expand(4,32,14,14).shape
7 Out[70]: torch.Size([4, 32, 14, 14])
8
9 In [72]: b.expand(-1,32,-1,-1).shape
10 Out[72]: torch.Size([1, 32, 1, 1])
11
12 In [71]: b.expand(-1,32,-1,-4).shape
13 Out[71]: torch.Size([1, 32, 1, -4])
14
```

repeat

Memory touched



```
1 In [74]: b.shape
2 Out[74]: torch.Size([1, 32, 1, 1])
3
4 In [75]: b.repeat(4,32,1,1).shape
5 Out[75]: torch.Size([4, 1024, 1, 1])
6
7 In [76]: b.repeat(4,1,1,1).shape
8 Out[76]: torch.Size([4, 32, 1, 1])
9
10 In [77]: b.repeat(4,1,32,32).shape
11 Out[77]: torch.Size([4, 32, 32, 32])
12
```

.t

```
1 In [78]: b.t()  
2 -----  
3 RuntimeError                                Traceback (most recent call last)  
4 <ipython-input-78-b510e0f64f40> in <module>()  
5 ----> 1 b.t()  
6  
7 RuntimeError: t() expects a 2D tensor, but self is 4D  
8  
9 In [79]: a=torch.randn(3,4)  
10  
11 In [80]: a.t()  
12 Out[80]:  
13 tensor([[ -0.4260,  -0.5704,  -0.5624],  
14         [  0.6139,   0.2291, -0.6666],  
15         [  1.2523,   0.5167,   0.1421],  
16         [  0.3274, -0.0412,   1.3652]])  
17
```

Transpose

```
1 [85]: a.shape # [4, 3, 32, 32]
2
3 In [86]: a1 = a.transpose(1,3).view(4,3*32*32).view(4, 3, 32, 32)
4 -----
5 RuntimeError                                Traceback (most recent call last)
6 <ipython-input-86-d78d8d9c8083> in <module>()
7 ----> 1 a1 = a.transpose(1,3).view(4,3*32*32).view(4, 3, 32, 32)
8
9 RuntimeError: invalid argument 2: view size is not compatible with input tensor's size
  and stride (at least one dimension spans across two contiguous subspaces). Call
  .contiguous() before .view(). at /opt/conda/conda-
  bld/pytorch_1544174967633/work/aten/src/TH/generic/THTensor.cpp:213
10
11 In [87]: a1 = a.transpose(1,3).contiguous().view(4,3*32*32).view(4, 3, 32, 32)
12
13 In [88]: a2 = a.transpose(1,3).contiguous().view(4,3*32*32).view(4, 32, 32,
  3).transpose(1, 3)
14
15 In [89]: a1.shape, a2.shape
16 Out[89]: (torch.Size([4, 3, 32, 32]), torch.Size([4, 3, 32, 32]))
17
18 In [91]: torch.all(torch.eq(a, a1))
19 Out[91]: tensor(0, dtype=torch.uint8)
20
21 In [92]: torch.all(torch.eq(a, a2))
22 Out[92]: tensor(1, dtype=torch.uint8)
```

permute

```
1 In [93]: a=torch.rand(4,3,28,28)
2
3 In [94]: a.transpose(1,3).shape
4 Out[94]: torch.Size([4, 28, 28, 3])
5
6 In [95]: b=torch.rand(4,3,28,32)
7
8 In [97]: b.transpose(1,3).shape
9 Out[97]: torch.Size([4, 32, 28, 3])
10
11 In [96]: b.transpose(1,3).transpose(1,2).shape
12 Out[96]: torch.Size([4, 28, 32, 3])
13
14 In [98]: b.permute(0,2,3,1).shape
15 Out[98]: torch.Size([4, 28, 32, 3])
16
```