创建Tensor

主讲人：龙良曲

Import from numpy

```
1 In [62]: a=np.array([2,3.3])
2
3 In [63]: torch.from_numpy(a)
4 Out[63]: tensor([2.0000, 3.3000], dtype=torch.float64)
5
6 In [65]: a=np.ones([2,3])
7 In [66]: torch.from_numpy(a)
8 Out[66]:
9 tensor([[1., 1., 1.],
10         [1., 1., 1.]], dtype=torch.float64)
```

Import from List

手写数据
大写+维度

```
 1 In [67]: torch.tensor([2., 3.2])
 2 Out[67]: tensor([2.0000, 3.2000])
 3
 4 In [68]: torch.FloatTensor([2., 3.2])
 5 Out[68]: tensor([2.0000, 3.2000])
 6
 7 In [69]: torch.tensor([[2., 3.2], [1.,
 8 Out[69]:
 9 tensor([[ 2.0000,  3.2000],
10          [ 1.0000, 22.3000]])
```

uninitialized

- Torch.empty()

- Torch.FloatTensor(d1, d2, d3)
  - NOT torch.FloatTensor([1, 2]) = torch.tensor([1, 2])

- Torch.IntTensr(d1, d2, d3)

uninitialized

```
 1 In [70]: torch.empty(1)
 2 Out[70]: tensor([0.])
 3
 4 In [71]: torch.Tensor(2,3)
 5 Out[71]:
 6 tensor([[ 3.1921e+27,  0.0000e+00, -1.0163e+11],
 7         [ 7.1186e-43,  0.0000e+00, -0.0000e+00]])
 8
 9 In [72]: torch.IntTensor(2,3)
10 Out[72]:
11 tensor([[ 1831143156,          0,  -776122816],
12         [        508,          0, -2147483648]], dtype=torch.int32)
13
14 In [73]: torch.FloatTensor(2,3)
15 Out[73]:
16 tensor([[ 3.1921e+27,  0.0000e+00, -8.0417e-17],
17         [ 7.1186e-43,  0.0000e+00, -0.0000e+00]])
```

set default type

```
1 In [74]: torch.tensor([1.2, 3]).type()
2 Out[74]: 'torch.FloatTensor'
3
4 In [75]: torch.set_default_tensor_type(torch.DoubleTensor)
5
6 In [76]: torch.tensor([1.2, 3]).type()
7 Out[76]: 'torch.DoubleTensor'
```

rand/rand_like, randint

- [0, 1]

- [min, max)

- *_like

```
1 In [77]: torch.rand(3,3)
2 Out[77]:
3 tensor([[0.1489, 0.3039, 0.0103],
4          [0.7305, 0.6398, 0.1361],
5          [0.0675, 0.8197, 0.0676]])
6
7 In [78]: a=torch.rand(3,3)
8
9 In [79]: torch.rand_like(a)
10 Out[79]:
11 tensor([[0.1823, 0.2776, 0.3376],
12          [0.2285, 0.7772, 0.9575],
13          [0.6914, 0.4166, 0.2171]])
14
15 In [80]: torch.randint(1,10,
16 Out[80]:
17 tensor([[8, 4, 2],
18          [1, 2, 7],
19          [3, 6, 2]])
```

randn

- N(0, 1)

- N(u, std)

```
 1 In [81]: torch.randn(3,3)
 2 Out[81]:
 3 tensor([[-0.7416, -1.7052, -0.1960],
 4         [ 0.9920,  0.4750,  0.7747],
 5         [-0.3542,  0.3421,  0.5126]])
 6
 7 In [90]: torch.normal(mean=torch.full([10],0), std=torch.arange(1, 0, -0.1))
 8 Out[90]:
 9 tensor([-1.7226,  1.0137,  0.7108, -0.4316,  0.4672, -0.4014,  0.5051,
10  0.0034, 0.2689, -0.2069])
11
12 In [91]: torch.normal(mean=torch.full([10],0), std=torch.arange(1, 0, -0.1))
13 Out[91]:
14 tensor([-1.0221, -1.3711,  1.0295,  0.1224,  0.3391,  0.1321, -0.6319,
15 -0.1722, 0.0483,  0.0629])
```

full

```
1 In [92]: torch.full([2,3],7)
2 Out[92]:
3 tensor([[7., 7., 7.],
4          [7., 7., 7.]])
5
6 In [94]: torch.full([],7)
7 Out[94]: tensor(7.)
8
9 In [95]: torch.full([1],7)
10 Out[95]: tensor([7.])
```

arange/range

```
1 In [96]: torch.arange(0,10)
2 Out[96]: tensor([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
3
4 In [97]: torch.arange(0,10,2)
5 Out[97]: tensor([0, 2, 4, 6, 8])
6
7 In [98]: torch.range(0,10)
8 C:\ProgramData\conda\Scripts\ipython:1: UserWarning: torch.range is deprecated in favor
  of torch.arange and will be removed in 0.5. Note that arange generates values in [start;
  end), not [start; end].
9 Out[98]: tensor([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
```

linspace/logspace

```
 1 In [99]: torch.linspace(0,10, steps=4)
 2 Out[99]: tensor([ 0.0000,  3.3333,  6.6667, 10.0000])
 3
 4 In [100]: torch.linspace(0,10, steps=10)
 5 Out[100]:
 6 tensor([ 0.0000,  1.1111,  2.2222,  3.3333,  4.4444,  5.5556,  6.6667,  7.7778,
 7          8.8889, 10.0000])
 8
 9 In [101]: torch.linspace(0,10, steps=11)
10 Out[101]: tensor([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9., 10.])
11
12 In [103]: torch.logspace(0,-1,steps=10)
13 Out[103]:
14 tensor([1.0000, 0.7743, 0.5995, 0.4642, 0.3594, 0.2783, 0.2154, 0.1668,
15 0.1292, 0.1000])
16
17 In [104]: torch.logspace(0,1,steps=10)
18 Out[104]:
19 tensor([ 1.0000,  1.2915,  1.6681,  2.1544,  2.7826,  3.5938,  4.6416,  5.9948,
20          7.7426, 10.0000])
```

Ones/zeros/eye

```
 1 In [105]: torch.ones(3,3)
 2 Out[105]:
 3 tensor([[1., 1., 1.],
 4         [1., 1., 1.],
 5         [1., 1., 1.]])
 6
 7 In [110]: torch.zeros(3,3)
 8 Out[110]:
 9 tensor([[0., 0., 0.],
10         [0., 0., 0.],
11         [0., 0., 0.]])
12
13 In [107]: torch.eye(3,4)
14 Out[107]:
15 tensor([[1., 0., 0., 0.],
16         [0., 1., 0., 0.],
17         [0., 0., 1., 0.]])
```

Ones/zeros/eye

```
 1 In [109]: torch.eye(3)
 2 Out[109]:
 3 tensor([[1., 0., 0.],
 4         [0., 1., 0.],
 5         [0., 0., 1.]])
 6
 7 In [111]: a=torch.zeros(3,3)
 8
 9 In [112]: torch.ones_like(a)
10 Out[112]:
11 tensor([[1., 1., 1.],
12         [1., 1., 1.],
13         [1., 1., 1.]])
14
```

randperm

- random.shuffle

```
1 In [127]: torch.randperm(10)
2 Out[127]: tensor([1, 5, 4, 2, 0, 6, 3, 9, 7,
   8])
```

```
1 In [113]: a=torch.rand(2,3)
2 In [114]: b=torch.rand(2,2)
3 In [115]: idx=torch.randperm(2)
4 In [116]: idx
5 Out[116]: tensor([1, 0])
6 In [122]: idx
7 Out[122]: tensor([0, 1])
8
9 In [123]: a[idx]
10 Out[123]:
11 tensor([[0.4283, 0.4819, 0.6252],
12          [0.9189, 0.7713, 0.9449]])
13
14 In [124]: b[idx]
15 Out[124]:
16 tensor([[0.2237, 0.6649],
17          [0.1008, 0.7560]])
18
19 In [125]: a,b
20 Out[125]:
21 (tensor([[0.4283, 0.4819, 0.6252],
22          [0.9189, 0.7713, 0.9449]]), tensor([[0.2237, 0.6649],
23          [0.1008, 0.7560]]))
```

下一课时

Tensor切片

Thank You.