创建

模型参数说明

```
opt.ouput_nc = 3       # number of output image channels: 3 for RGB and 1 for grayscale
opt.ndf = 64                 # number of discriminator filters in the first conv layer
opt.netD = 'basic'           # The basic model is a 70x70 PatchGAN. (论文所用的模型)
opt.n_layers_D = 3           # only used if netD==n_layers
opt.norm = instance        # instance normalization or batch normalization [instance | batch | none]
opt.init_type = normal       # network initialization [normal | xavier | kaiming | orthogonal]
opt.init_gain = 0.02         # scaling factor for normal, xavier and orthogonal
```

命令

```
 netD = networks.define_D(opt.output_nc, opt.ndf, opt.netD,opt.n_layers_D, opt.norm,
     opt.init_type, opt.init_gain)
```

模型结构

```
NLayerDiscriminator(
(model): Sequential(

(0): Conv2d(3, 64, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
(1): LeakyReLU(negative_slope=0.2, inplace)
(2): Conv2d(64, 128, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
(3): InstanceNorm2d(128, eps=1e-05, momentum=0.1, affine=False, track_running_stats=False)
(4): LeakyReLU(negative_slope=0.2, inplace)
(5): Conv2d(128, 256, kernel_size=(4, 4), stride=(2, 2), padding=(1, 1))
(6): InstanceNorm2d(256, eps=1e-05, momentum=0.1, affine=False, track_running_stats=False)
(7): LeakyReLU(negative_slope=0.2, inplace)
(8): Conv2d(256, 512, kernel_size=(4, 4), stride=(1, 1), padding=(1, 1))
(9): InstanceNorm2d(512, eps=1e-05, momentum=0.1, affine=False, track_running_stats=False)
(10): LeakyReLU(negative_slope=0.2, inplace)
(11): Conv2d(512, 1, kernel_size=(4, 4), stride=(1, 1), padding=(1, 1))

)
)
```