计14 汪隽立 2021012957

ANN-hw4

0. 参数选择

如没有特殊说明，除了`latent_dim`与`hidden_dim`以外的超参数均选择默认值，即：

```
batch_size = 64
num_training_steps = 5000
learning_rate = 0.0002
beta1 = 0.5
```

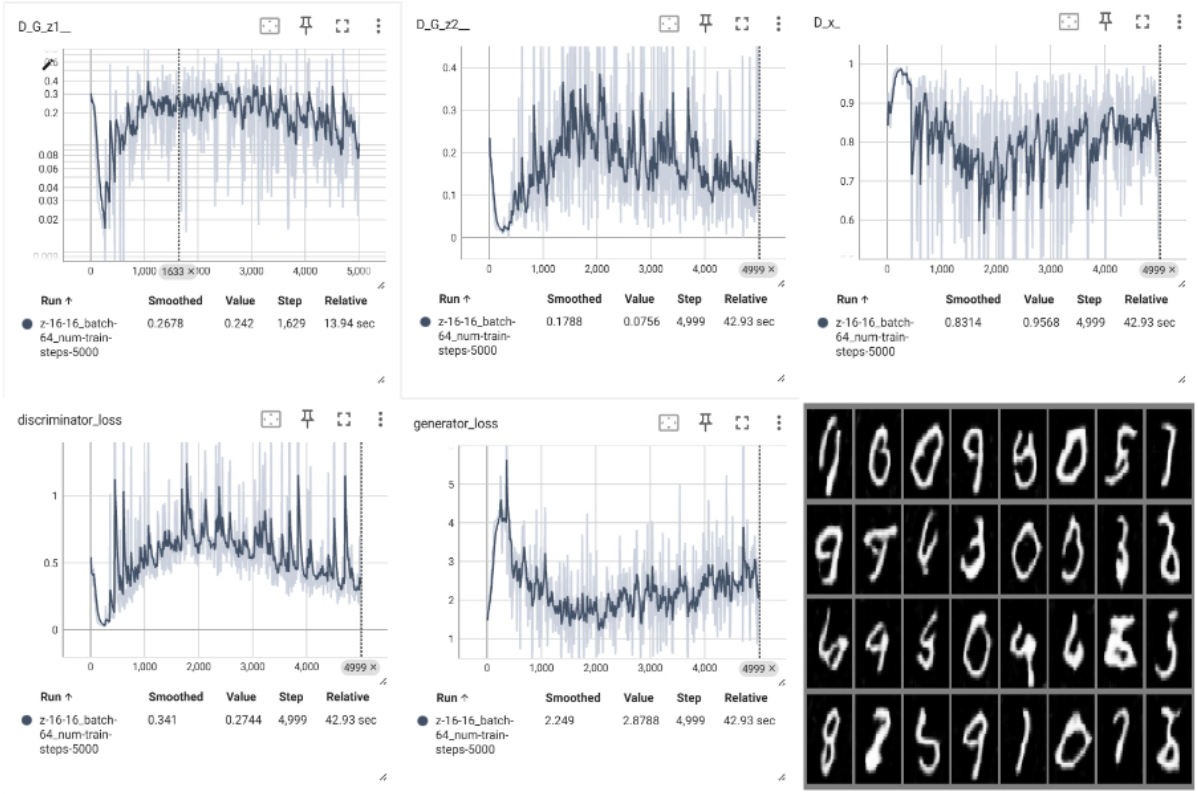
1. 改变`latent_dim`和`hidden_dim`，展示训练曲线和生成图片

在实验过程中，latent_dim = {16, 32, 64, 100}，hidden_dim = {16, 32, 64, 100}，共有$4 \times 4 = 16$种组合。为节省报告空间，在这一小节仅展示

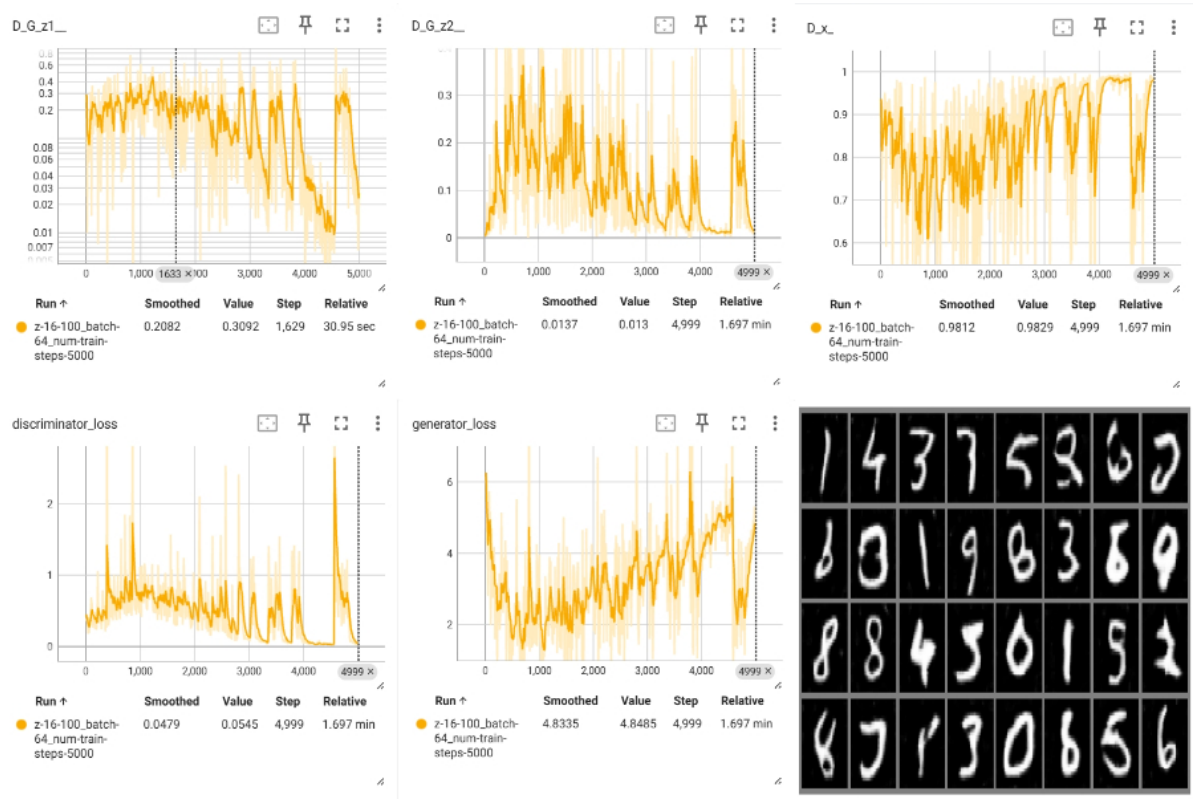
```
latent_dim = 16, hidden_dim = 16
latent_dim = 16, hidden_dim = 100
latent_dim = 100, hidden_dim = 16
latent_dim = 100, hidden_dim = 100
```

四种设定下的训练曲线和生成图片。由于曲线振荡比较严重，在tensorboard中采用了smoothing=0.75

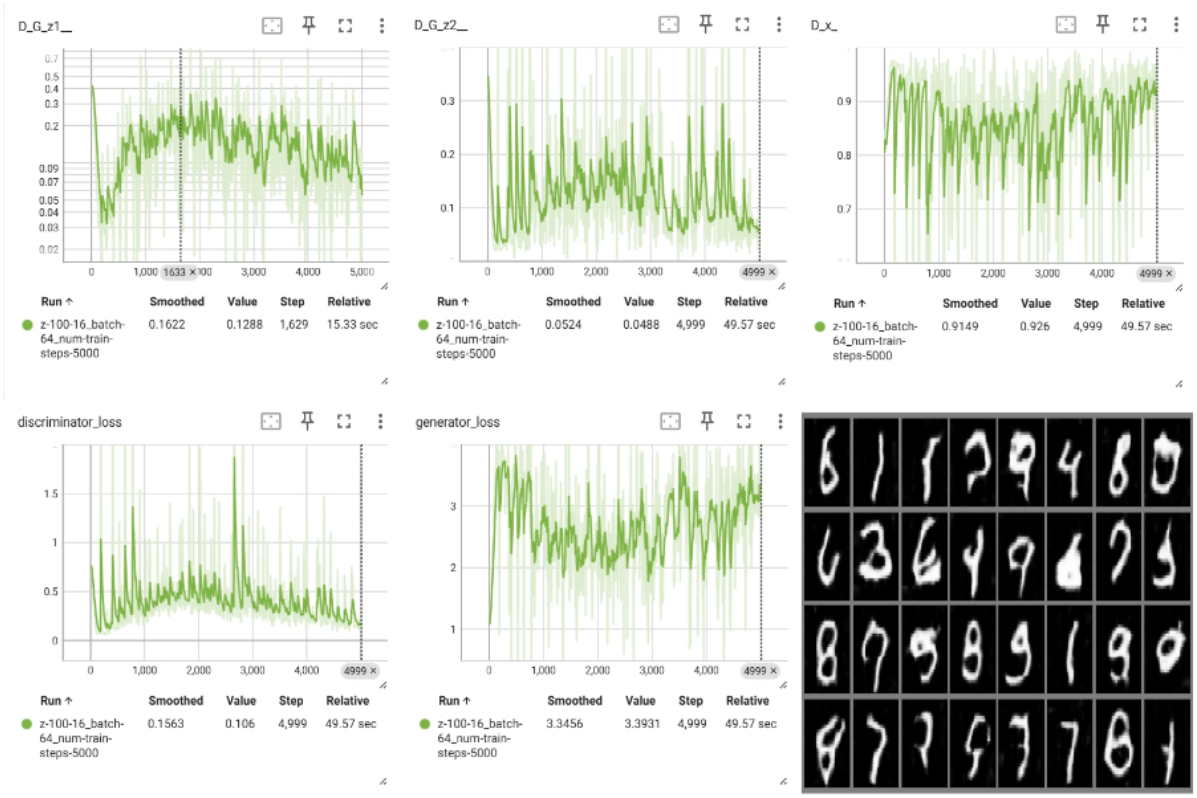
- latent_dim = 16, hidden_dim = 16



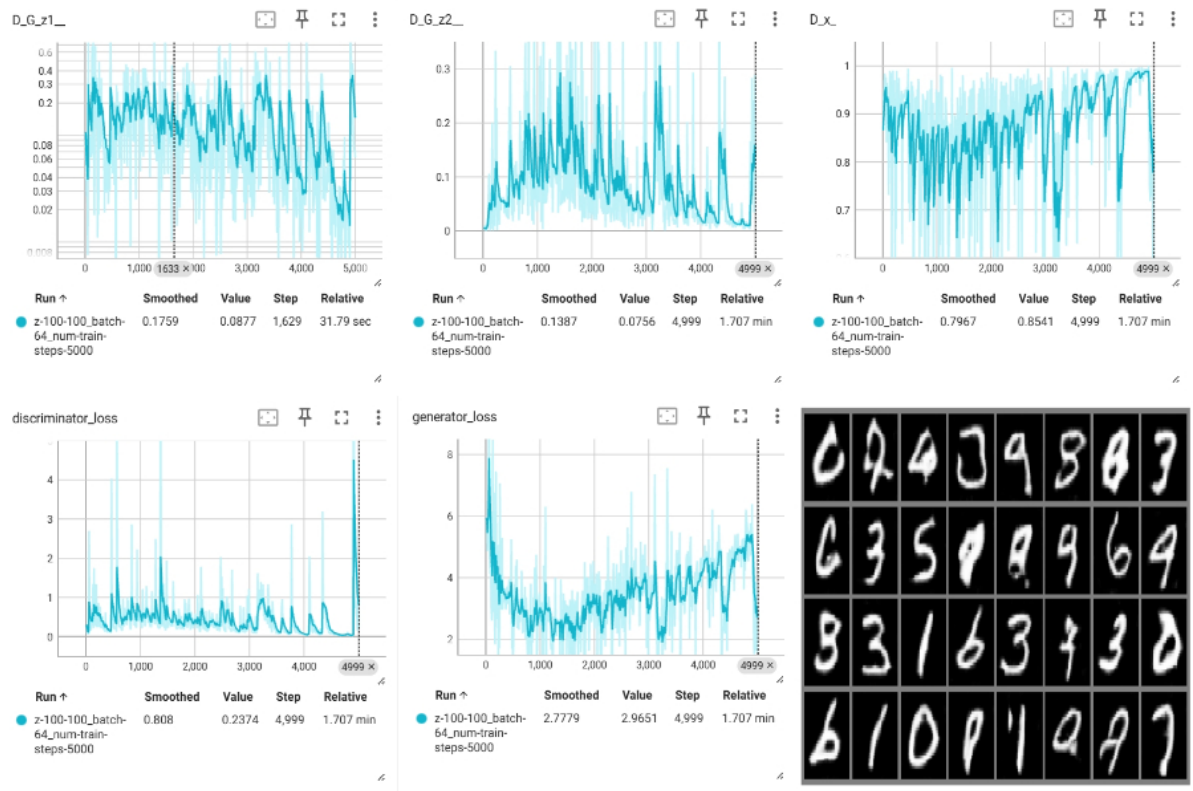
- latent_dim = 16, hidden_dim = 100



- latent_dim = 100, hidden_dim = 16



- latent_dim = 100, hidden_dim = 100



2. FID_Score

除了以上展示的4组实验以外，我们展示进行的所有16组实验的FID_Score

	latent_dim=16	latent_dim=32	latent_dim=64	latent_dim=100
hidden_dim=16	82.43	60.84	77.4	83.56
hidden_dim=32	84.07	77.85	51.43	88.27
hidden_dim=64	44.36	34.86	60.5	45.16
hidden_dim=100	68.19	38.83	43.21	**24.72**

最好一组的设定为latent_dim = hidden_dim = 100。

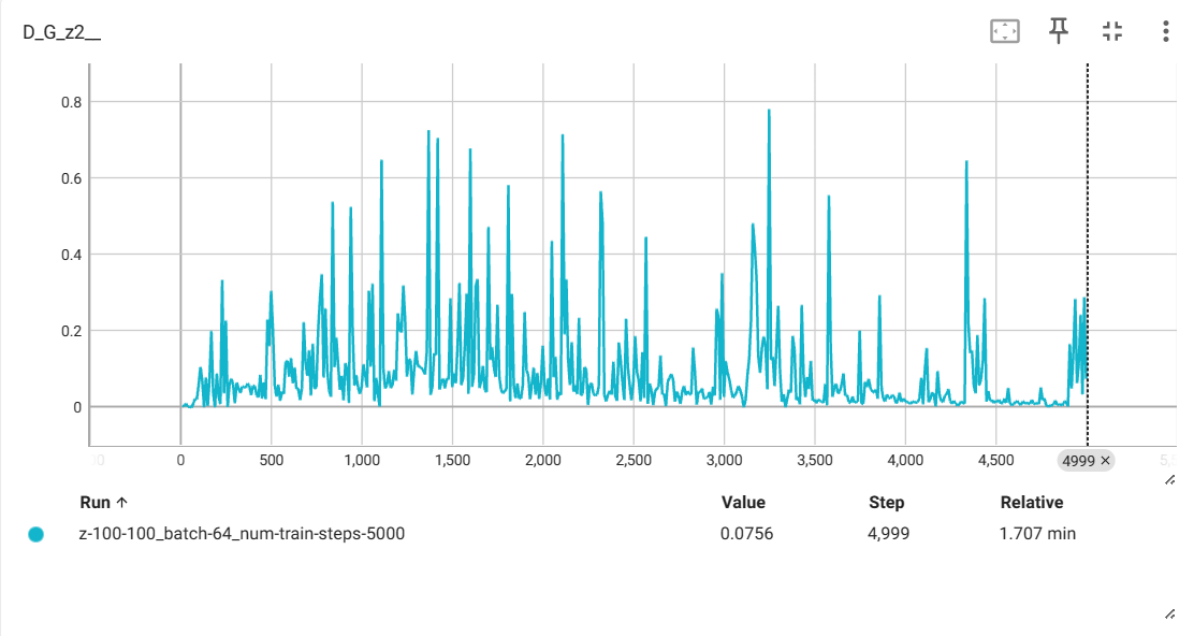
3. latent_dim与hidden_dim对GAN的影响

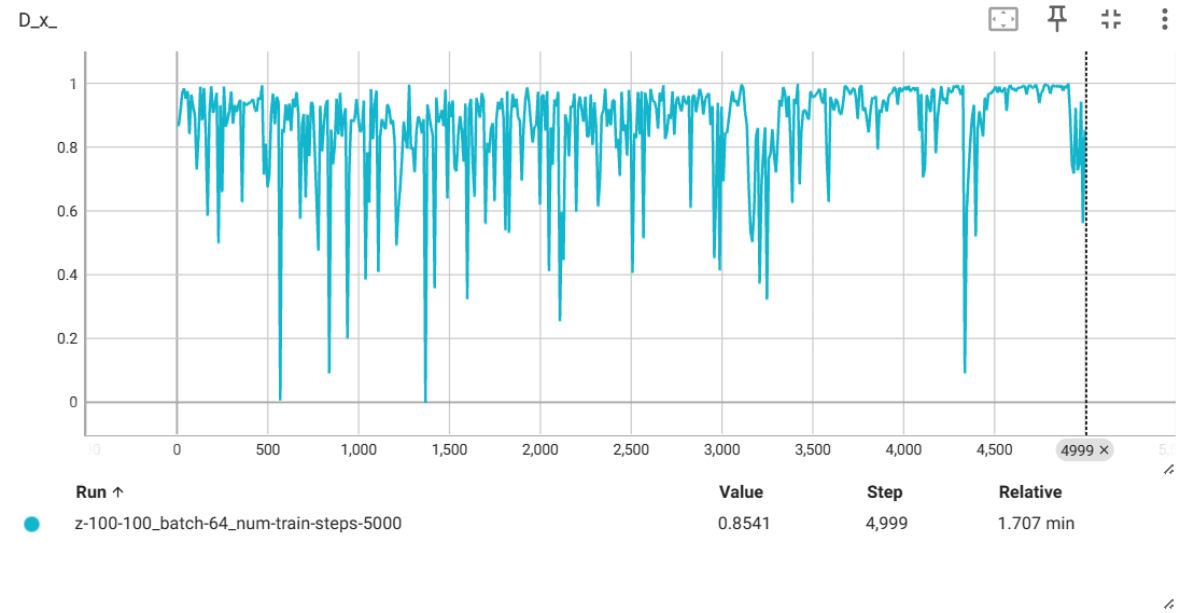
- latent_dim
  - latent_dim代表了随机噪声向量z的维数，latent_dim的增加代表了随机噪声表示能力的增加。
  - 在我们的实验中，对于比较小的hidden_dim (16, 32, 64)，latent_dim对实验结果没有显著的趋势影响，这可能是因为latent_dim=16已经能够比较好地泛化噪声的各种情况，故再增加latent_dim对模型能力没有显著提升。
  - 但在hidden_dim=100这组实验中，latent_dim=16与latent_dim=100两组实验产生了比较大的差异。这可能是因为更大的latent space导致需要更大的假设空间，故更大的参数量（体现为更大的hidden_dim）在此时便能发挥更好的结果。
- hidden_dim
  - hidden_dim代表了反卷积网络中的通道数，hidden_dim的增加代表了Generator和Discriminator参数的增加。

- 在我们的实验中，基本呈现了模型能力随着hidden_dim增加而提升的趋势。这可能是因为假设空间还没有达到足够大的情况，模型的能力随着参数的增加正在不断提升。

4. 观察D的判别策略，是否达到了纳什均衡?

以latent_dim = 100, hidden_dim = 100的实验组作为观察对象，观察D_G_z和D_x：



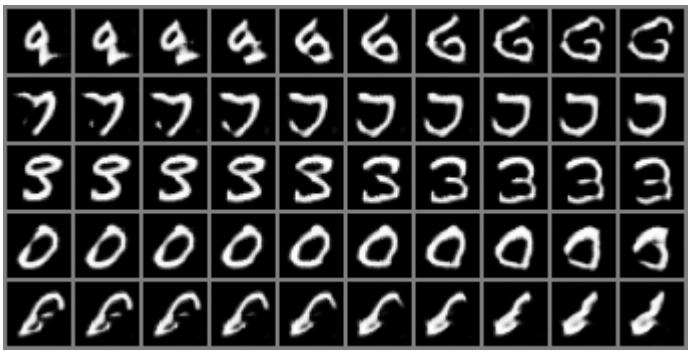


- 可以发现：Discriminator总是给生成的图像较低的分数（大约0.1到0.2之间），给真实的图像较高的分数（大约0.8以上）。
- 然而，GAN最终要达到的纳什均衡为，G能够完全欺骗D，即D给G_z的分数应当在0.5附近。
- 故可以得出结论，我们的模型没有达到纳什均衡。一个可能的原因是，D的能力在学习过程中提升的比G要多更多，导致了G的能力不足以欺骗D。

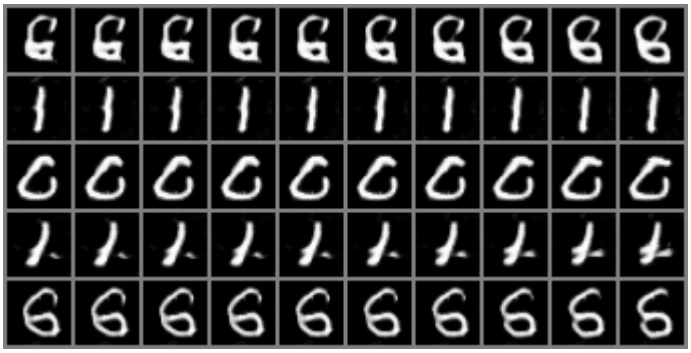
5. 内插值与外插值

选取latent_dim = 100, hidden_dim = 100进行实验

- 内插值结果：



- 外插值 (i = {-K, -K+1, ..., 0}):



- 外插值 (i = {K, K+1, ..., 2K})

可以看到，不论是外插值还是内插值，对于连续变化的噪声z，Generator能将其映射成相近的图像。这说明了G比较好地学习到了Latent Space的特征，使得$G(\cdot)$对于Latent Space是一个连续的函数。

6. 模式坍塌

选取latent_dim = 100, hidden_dim = 100进行实验。生成一百个数字:

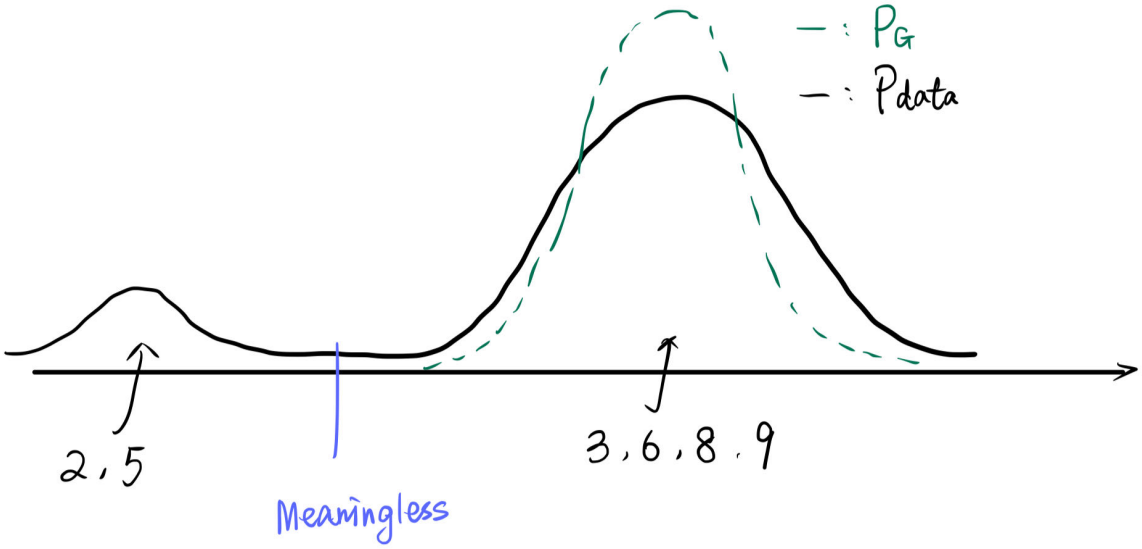
其对应的最接近的数字为：

0	7	9	3	1	7	9	9	8	0
6	0	3	4	1	3	0	9	9	5
8	8	8	2	9	6	9	7	0	0
5	6	5	8	6	9	3	4	5	1
6	7	7	5	0	8	5	9	8	7
9	7	9	3	7	1	7	6	8	8
1	6	6	0	8	6	0	3	3	0
8	6	9	9	9	6	3	0	9	0
3	7	9	0	7	8	0	3	3	6
8	3	3	9	8	8	8	0	2	0

0	1	2	3	4	5	6	7	8	9
16	5	2	13	2	6	12	11	16	17

可以发现，3，6，8，9在生成的数据中明显多于10%，而1，2，4，5在生成的数据里明显少于10%。而实际上p_{data}中各数字的占比都为10%，故我们的模型确实发生了模式坍塌。

- 可能的原因是，由于3，6，8，9特征比较接近，分布可能是如下图所示的样子：



- 另一个原因是，由于D的判别能力要远远超出G的欺骗能力，而3，6，8，9的特征比较接近（都有圆弧或圆圈），故G为了提高自己的目标函数，最终使得p_G中这几个数字出现概率更大。

7. 消融实验

选取latent_dim = 100, hidden_dim = 100进行实验。

首先，我们展示原模型生成的100张图片：

- MLP实验

由于MLP的参数量更大，我们将G的decoder结构设置为：

```
self.decoder = nn.Sequential(
    nn.Linear(in_features=latent_dim, out_features=1024),
    nn.BatchNorm1d(num_features=1024),
    nn.ReLU(),
    nn.Linear(in_features=1024, out_features=512),
    nn.BatchNorm1d(num_features=512),
    nn.ReLU(),
    nn.Linear(in_features=512, out_features=256),
    nn.BatchNorm1d(num_features=256),
    nn.ReLU(),
    nn.Linear(in_features=256, out_features=32 * 32),
    nn.Tanh()
)
```

D的clf结构为：

```
self.clf = nn.Sequential(
    nn.Linear(num_channels * 32 * 32, 1024),
    nn.LeakyReLU(0.2, inplace=True),
    nn.Linear(1024, 256),
    nn.LeakyReLU(0.2, inplace=True),
    nn.Linear(256, 64),
    nn.LeakyReLU(0.2, inplace=True),
    nn.Linear(64, 1),
    nn.Sigmoid()
)
```

得到的FID score为152.645，生成的100张图片如下：

从个人感觉的角度来说，生成的图片质量反而要更好了，至少可辨认了许多（虽然FID score更高）。可能的原因是，对于MLP来说，我们的训练轮次已经可以让它具有不错的泛化能力。

- 去除LayerNorm2d

得到的FID score为24.961，生成结果如下：

虽然图像的可辨认度有所提升，但可以感受到模式坍塌更严重了一些。

- 总的来说，对我们最好的模型而言，将decoder改用MLP会导致FID score提升非常多，而且生成**一种风格完全不同的图像**。而将LayerNorm2d去除则对FID score有一定的提升，对图像的生成风格等没有太大改变。总的来说，将decoder改为MLP的影响≫去除LayerNorm2d的影响。