油气人工智能基础及应用第九周作业

姓名：付楷涵 学号：2022211037 班级：研22-1班

# Generative Adversarial Nets(GAN)原理

为了弄清楚GAN的原理，在听了b站李沐关于GAN论文的讲解视频后又精读了原论文，以下是论文精读内容：

## 摘要(Abstract)

我们通过相互对抗的过程(adversarial process)提出了一个新的估计生成模型(estimating generative models)**框架(**framework**)**。要同时训练两个模型：**生成模型(**generative model**)**和**判别模型(**discriminative model**)**。其中生成模型是要**捕捉训练数据的分布**，判别模型是**估计一个样本是来自训练数据集而不是生成模型的概率**。对的训练过程是**最大化犯错误的概率**。对于任何和的函数空间中，都存在唯一解，使得可以复原训练数据的分布同时恒等于(在理论推导中会解释)。若和都是多层感知机(multilayer perceptron, MLP),则系统可以通过反向传播(backpropagation)来训练,整个过程不需要任何马尔科夫链。

核心就是两个模型和，只要弄明白两个模型代表的含义，就能知道这个算法的思想，而后就转化为一个数学层面的最优化问题。

## 简介(Introduction)

首先作者给深度学习下了一个定义：**深度学习的目的是发现丰富的、有层次的模型来表示在人工智能应用中遇到的各种数据的概率分布。**作者提到目前为止，深度学习通过反向传播与丢弃算法(backpropagation and dropout algorithms) 在**判别模型**上有着惊人的成功，而对于深度**生成模型**逐渐失去了影响，由于通过极大似然估计来判别数据集的分布参数太过复杂。

接下来作者用一个很形象的例子来说明对抗网络模型是在干什么：在所提出的对抗网络框架中，生成模型与对手——判别模型进行竞赛，用来确定样本是来自还是来自训练数据。生成模型可以被认为类似于一组造假者(counterfeiters)，试图生产假币并在未经检测的情况下使用，而辨别模型则类似于警察(police)，试图检测假币。在这个游戏中的竞争促使两队改进他们的方法，直到仿冒品与真品难以区分。

该框架可以为多种模型和优化算法生成特定的训练算法。在本文中，我们探讨了生成模型将**随机噪声通过多层感知器来生成样本**的特殊情况，而判别模型也是多层感知机。在这种情况下，我们可以仅使用非常成功的反向传播和丢弃算法来训练这两个模型，并仅使用正向传播从生成模型中采样,不需要统计推断或马尔可夫链。

## 对抗网络(Adversarial nets)

对抗生成模型最简单的应用是当生成模型和判别模型都是多层感知机。为了通过训练数据来学习生成器(generator)的分布,我们输入噪声随机变量,用来代表到训练数据集样本空间的映射(mapping)，其中是一个代表参数为的多层感知机的可微函数。 同样是一个多层感知机函数，其**代表数据来自训练样本而不是生成器的概率。**于是我们可以定义目标函数：

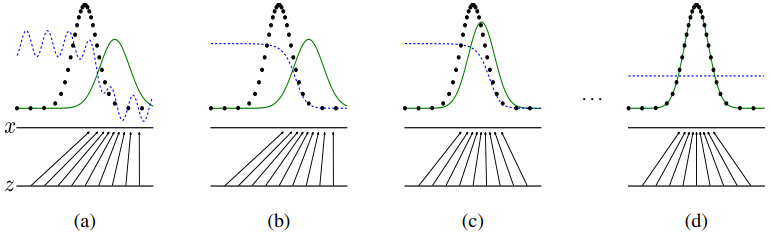
   对于警察，若来自训练样本，则应增大，进而增大；若来自生成器，则应增大，进而增大。由此，判别模型的目标函数为：

对于造假者，若来自训练样本，则应增大，进而减小，进而减小。由此，生成模型的目标函数为：

进而得到整个模型的目标函数：

### 理论推导(Theoretical Results)

当噪声随机变量时，生成模型通过样本分布**隐式地**定义了一个概率分布。下图可以很好的理解整个训练过程：



通过同时更新辨别模型分布（，蓝色虚线）来训练GAN，使得其区分来自训练数据分布的样本（黑色虚线）和来自生成模型分布的样本（绿色，实线）。下面的横线是噪声随机变量的采样区域，图示是均匀采样。上面的横线是域的一部分。而箭头代表了映射是如何迫使非均匀分布转换为训练样本

(a)对于如图所示的对抗：与相似，同时判别模型是部分准确的;

(b)通过对判别模型的训练，其逐渐的可以区分生成数据分布和训练数据分布，其最优值收敛到;

(c)通过更新迭代生成模型,其生成的数据与训练样本更加接近;

(d)通过几次的迭代，最终和将达到而无法在进行提升,同时判别模型满足而无法进行区分。

数学优化模型推导：

对于求解我们首先固定参数,则此时中的期望可以转化为积分，即：

我们想求的最大值，则令其导数等于0：

由此得到的最优值：

此时，我们将的最优值带入到中继续求解目标函数值：

这里用到了一个信息论中的KL散度，我简单看了一下，大概是一个概率的描述定义，这里利用KL散度进行求解：

这里将分母除以2的目的是由于的范围是[0,2]，不属于随机变量，因此将其范围压缩到[0,1]，只有分子分母均为随机变量才可以利用KL散度求解。上式转化为：

易知当时等号成立，由此，的最优解为：，同时.

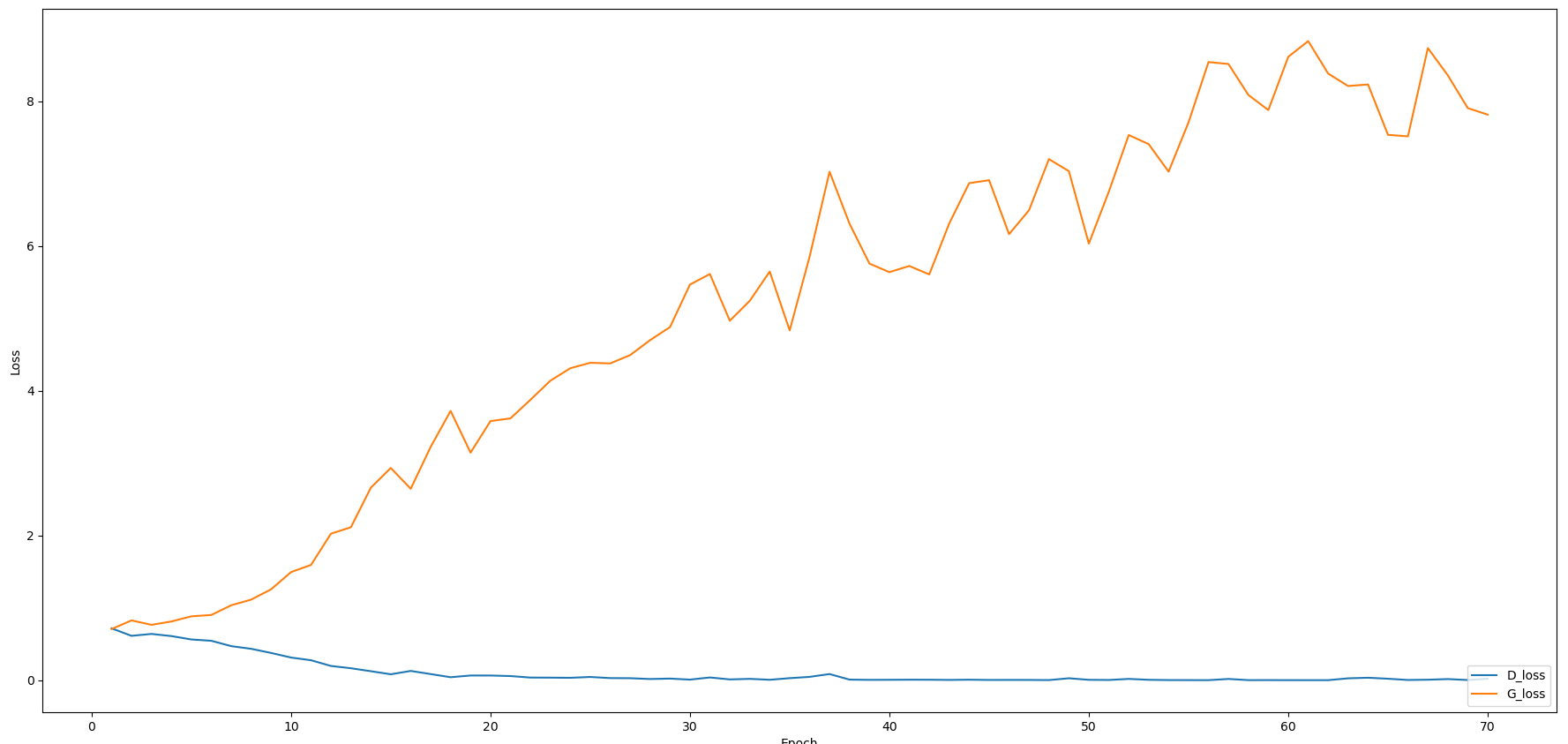
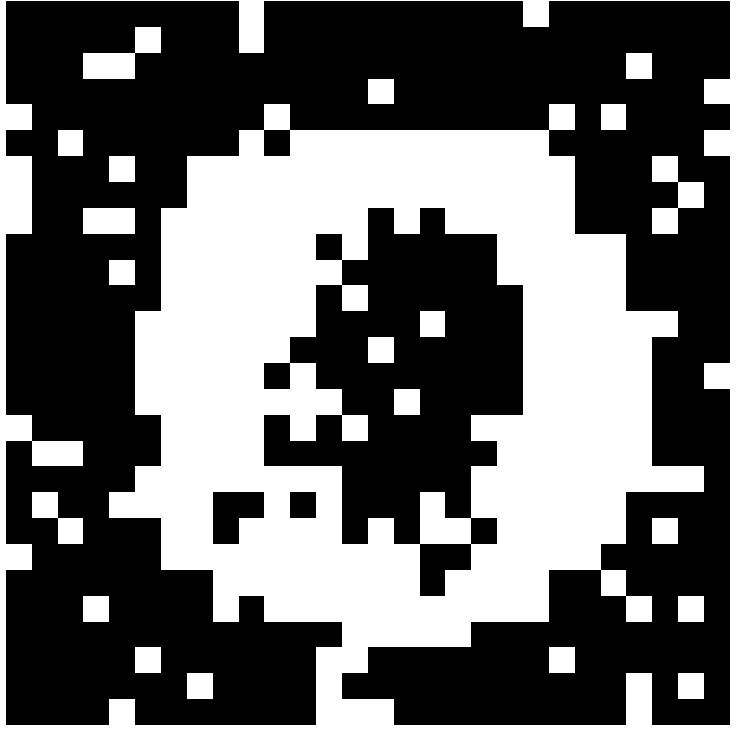
### 评价与结论(Conclusions)

 作者在第六段说明了模型目前存在的优缺点，作者当时提出来时其实模型效果并不好，但是基本可以达到效果，即这个模型提出的是“初号机”，目前通过很多研究者的改良，GAN模型已经可以达到很好的效果了，在此简单的写一下文章中提出的优缺点：  
  缺点是不能显式的将表示出来（但我寻思他的创新点不就是绕过了显式表达吗...）;而优点就是他说了很多遍的不依赖于马尔科夫链。  
  最后，作者给出了模型的实验效果，可以看到除了生成数字其余的效果并不是很好。

# 程序实现效果最好的图

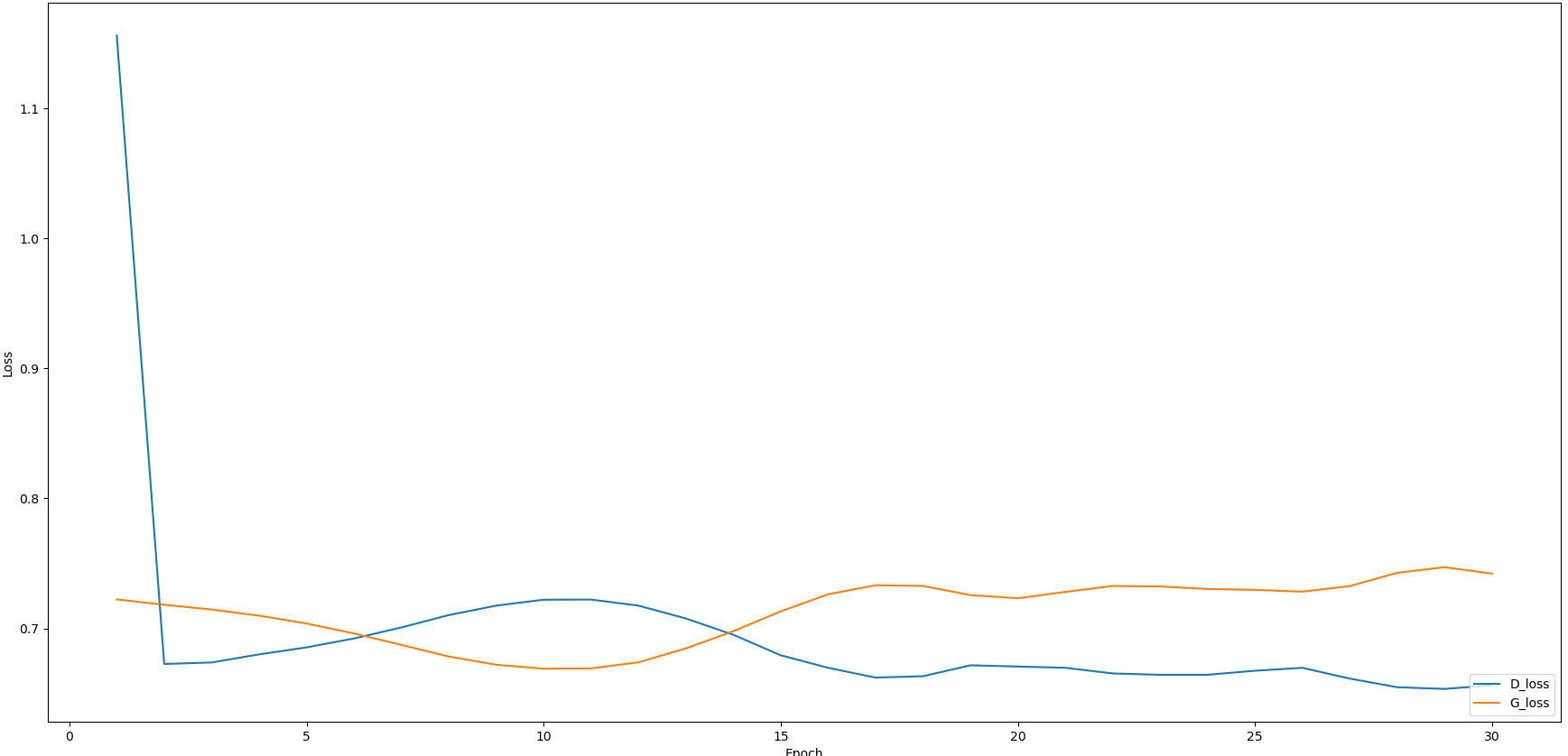
通过控制变量的方法来调整各个参数（第三章参数影响），我们对每个参数进行逐一调整，最终得到的目前效果的最好的图的参数如下：

最终效果如下：



# 改变参数程序结果的变化

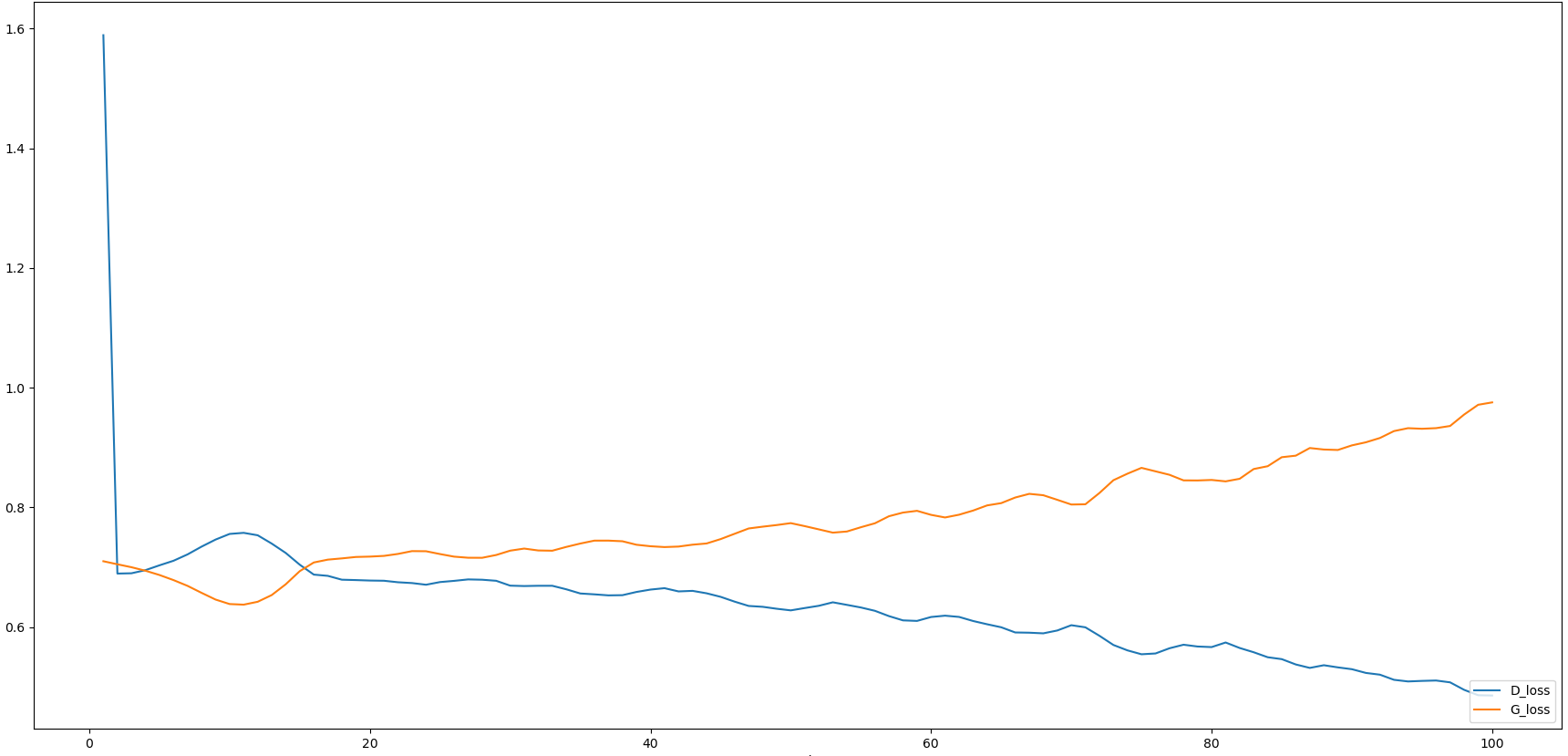
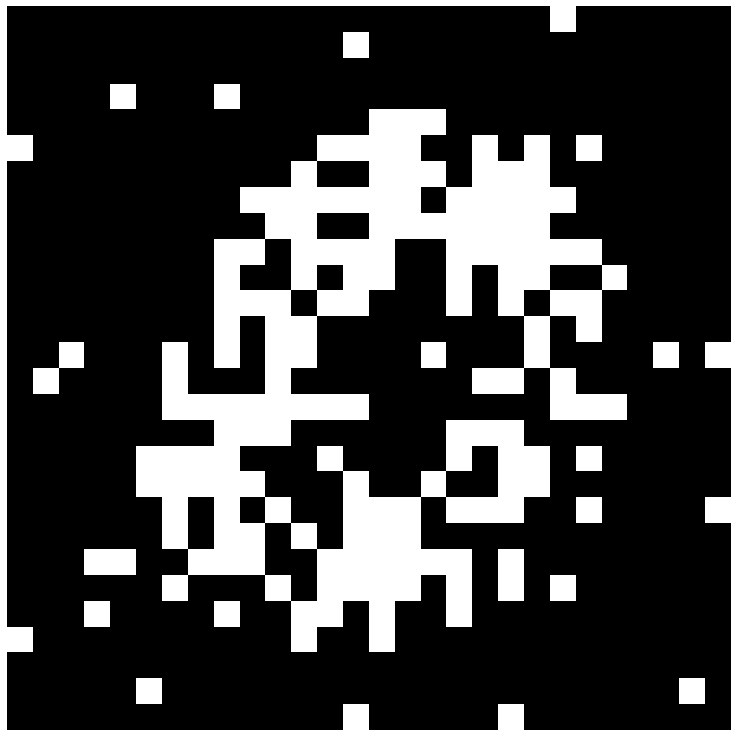
原始参数：epoch=30 lr=0.00001 batchsize=228 input\_size=200 原始参数得到的结果如下图所示：



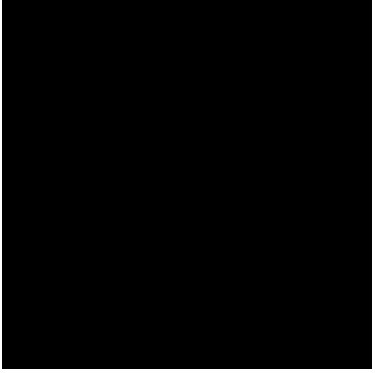
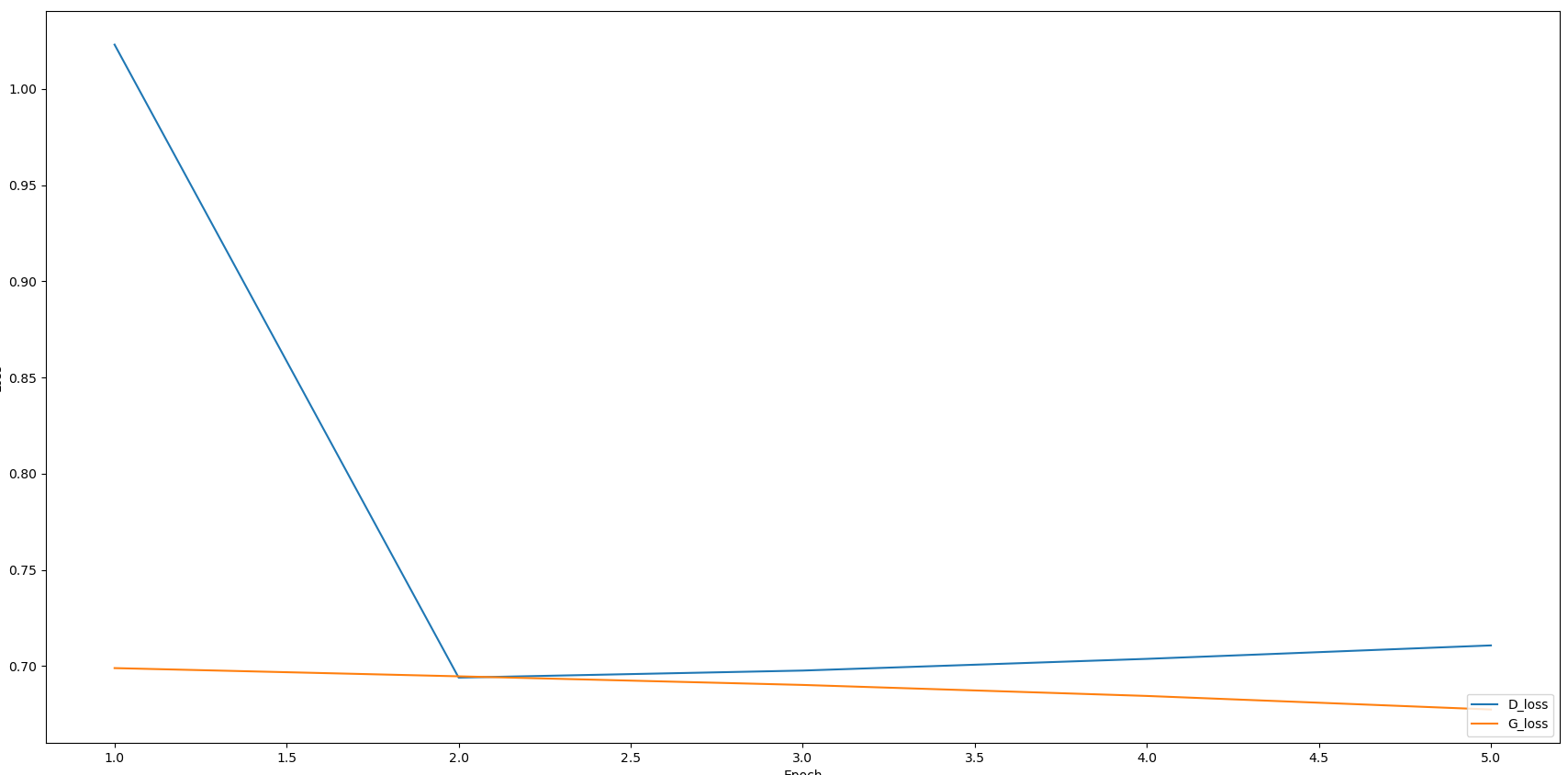
现对四种参数依次进行调整：

1. **调整epoch**

若epoch过大(epoch=100)，在没有GPU加速的情况下(4it/s)，运行时间大大增加，生成器的损失逐渐增加，辨别器的损失逐渐减少，最后的生成效果似乎比原始图片有所提升，效果如下图所示：

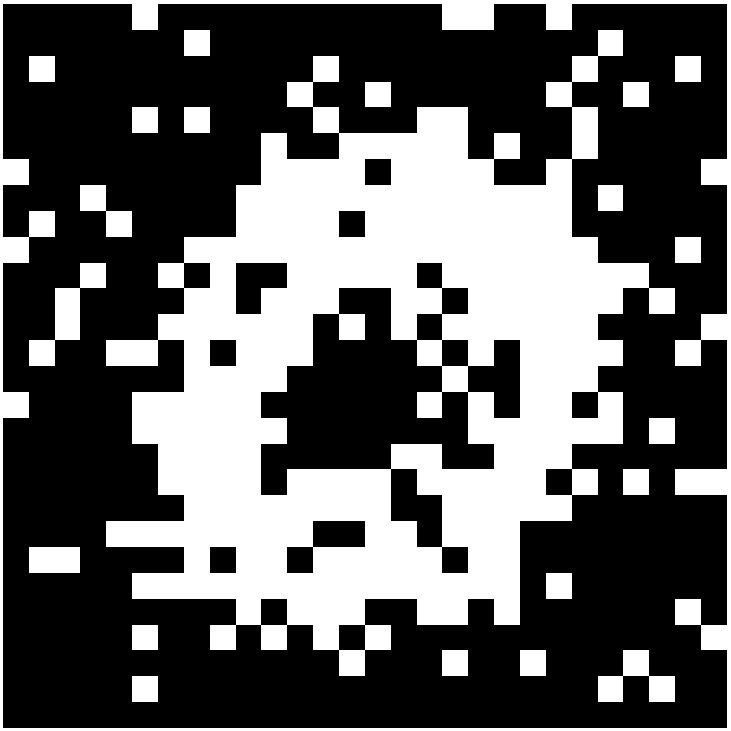
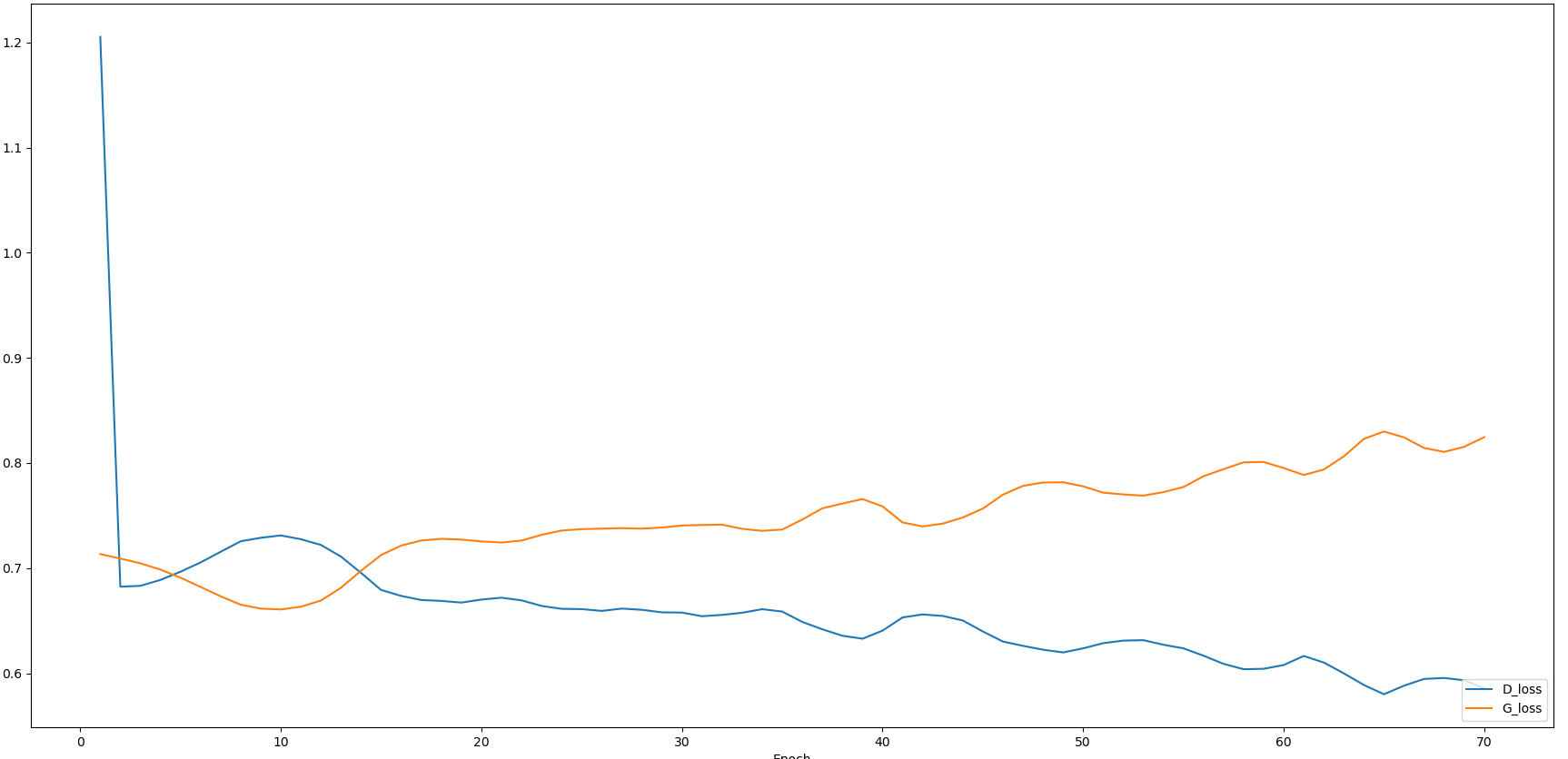


若epoch过小(epoch=5)，在没有GPU加速的情况下(4it/s)，运行时间非常短，程序崩塌没有任何图片生成，效果如下：

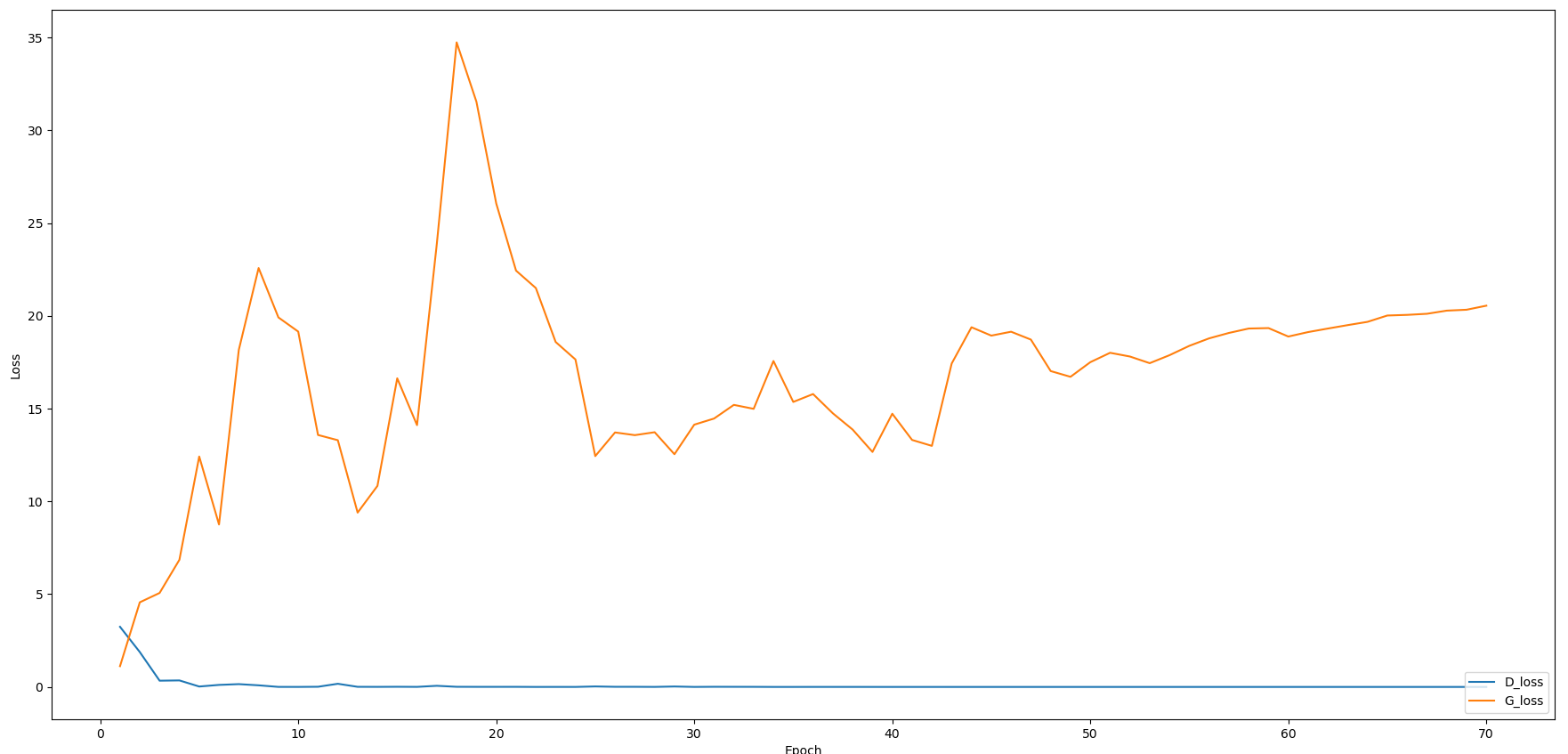
结论：就epoch而言，增加epoch对最后的生成图片效果是有帮助的，由于没有GPU的加速，当epoch=100时所耗费的时间就非常多，因此综合来看，较好的epoch范围应该为60-80。

例如epoch=70时的效果图：

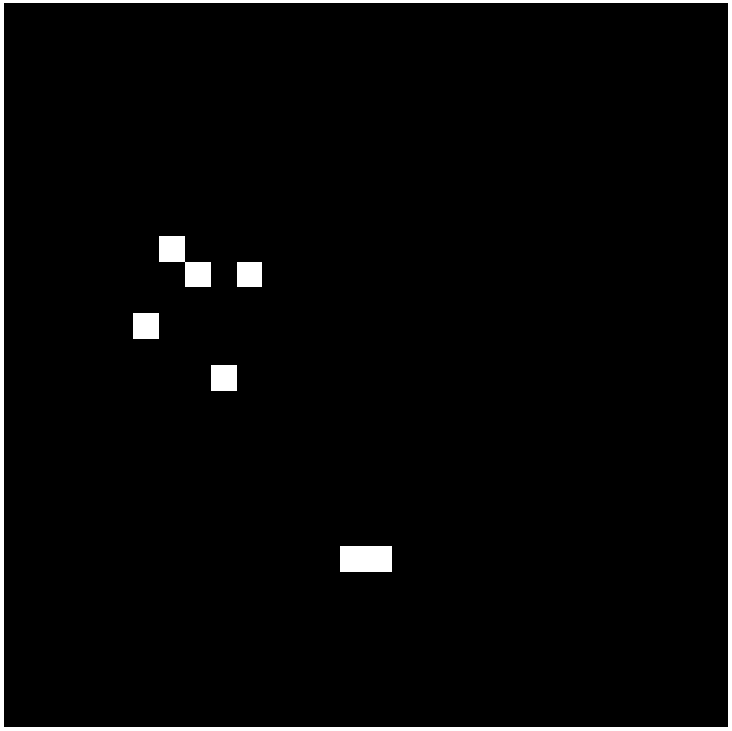
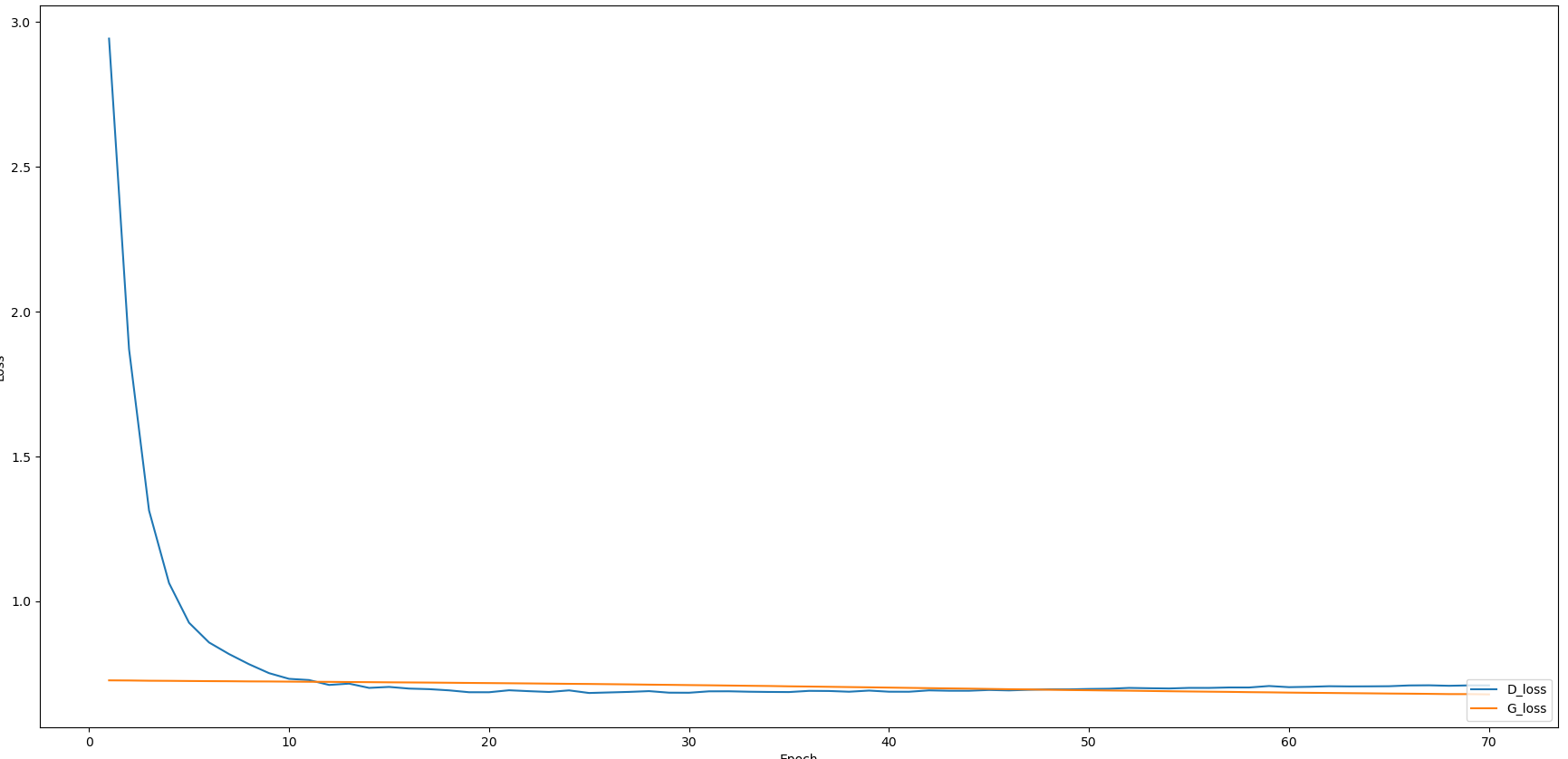
 

1. **调整lr（固定epoch=70）**

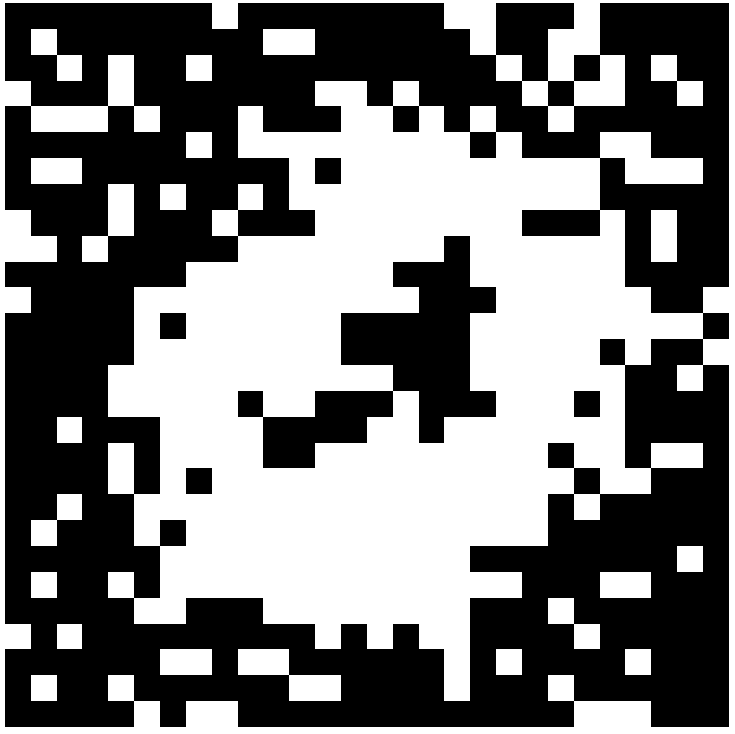
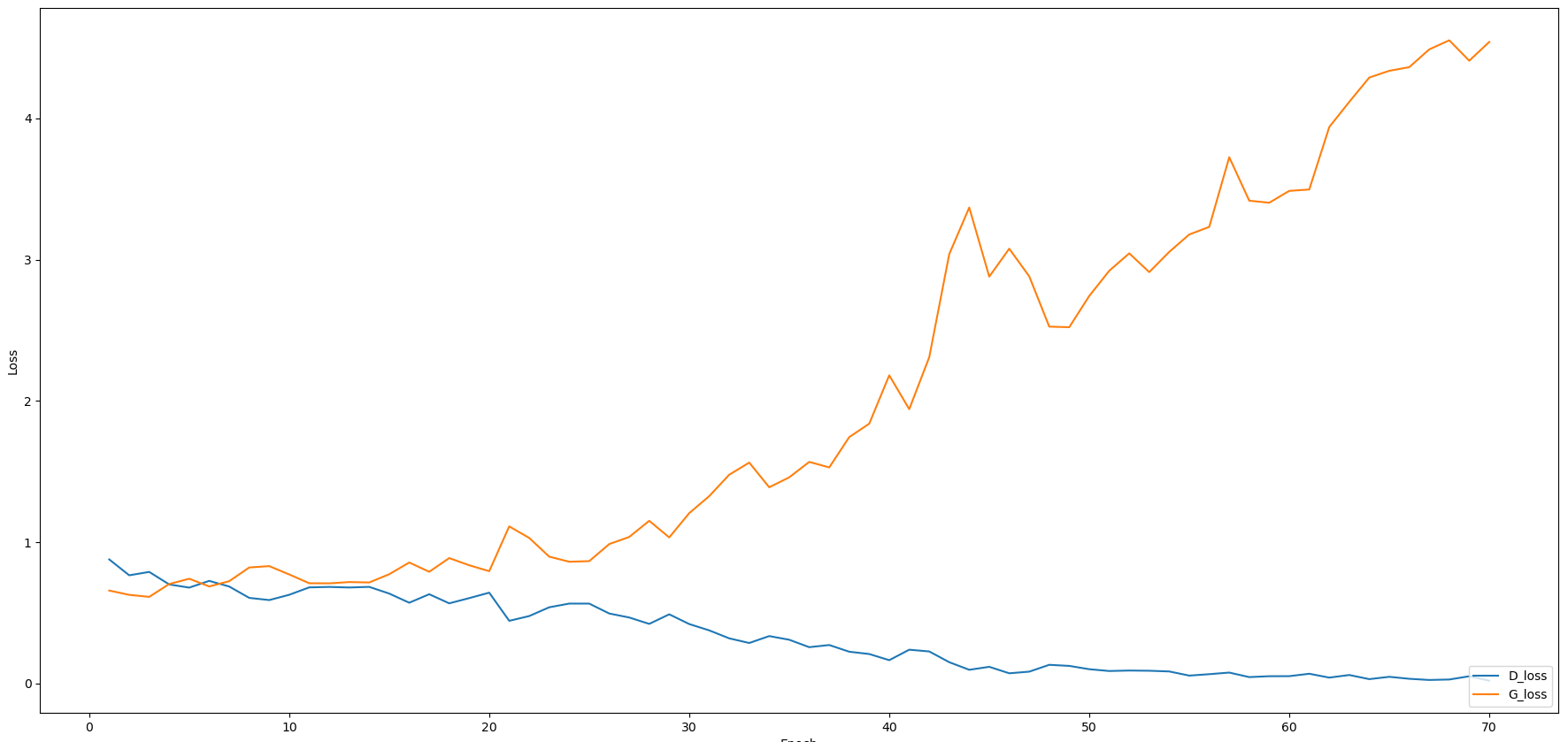
采用与上述相同的测试方法，若增加lr(lr=0.001)，在没有GPU加速的情况下(4it/s)，运行时间无比的长，最后的生成效果很差，效果如下图所示：

若减小lr (lr=0.00001)，最后没有生成图片，效果如下图所示：

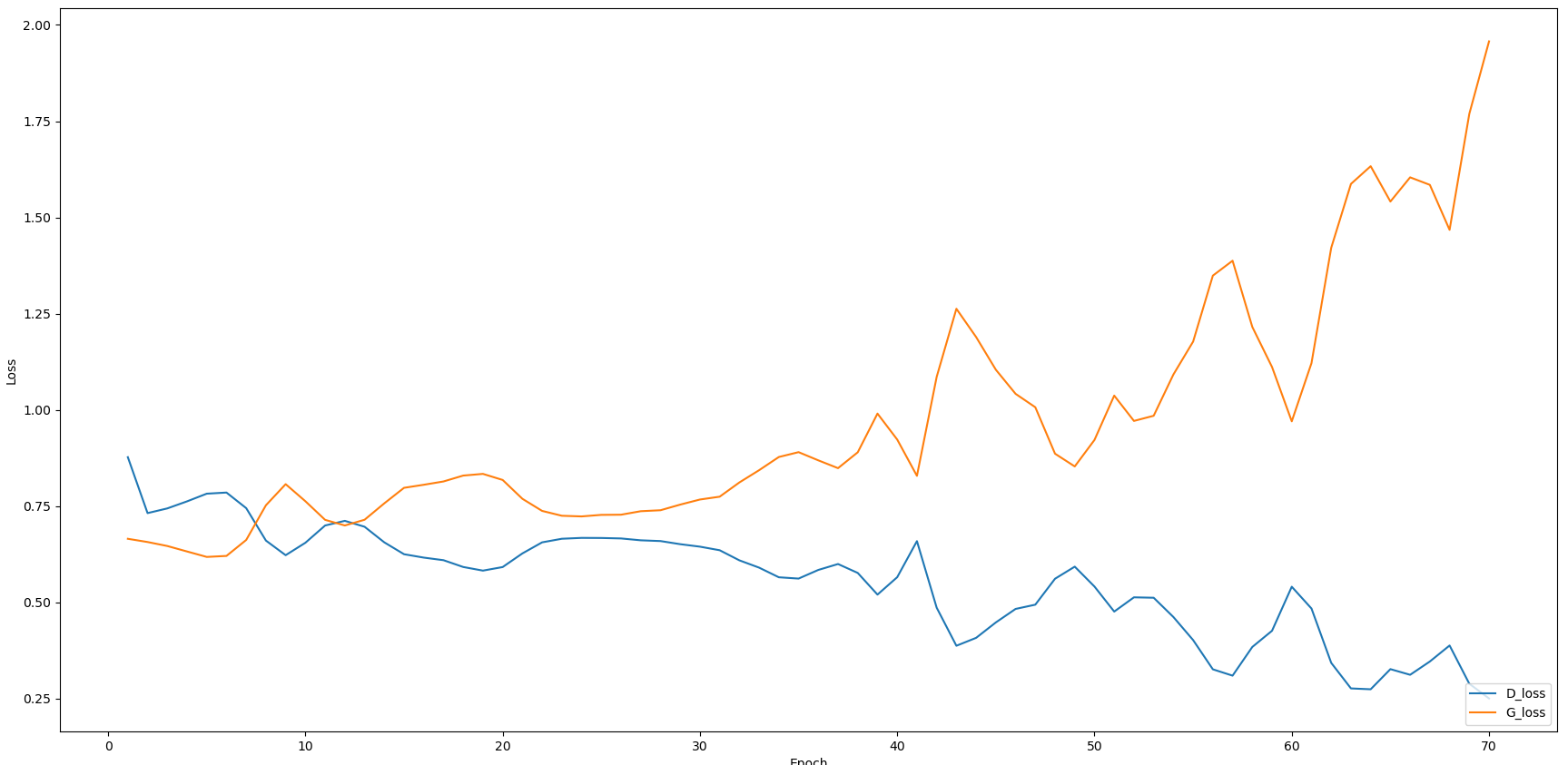
 

结论：就lr而言，无论步长选取的过大还是过小对于结果均产生较大的不利影响，因此确定一个较好的步长对实验效果有着非常重要的影响，在此小幅度的提升步长观察结果(lr=0.00005)：

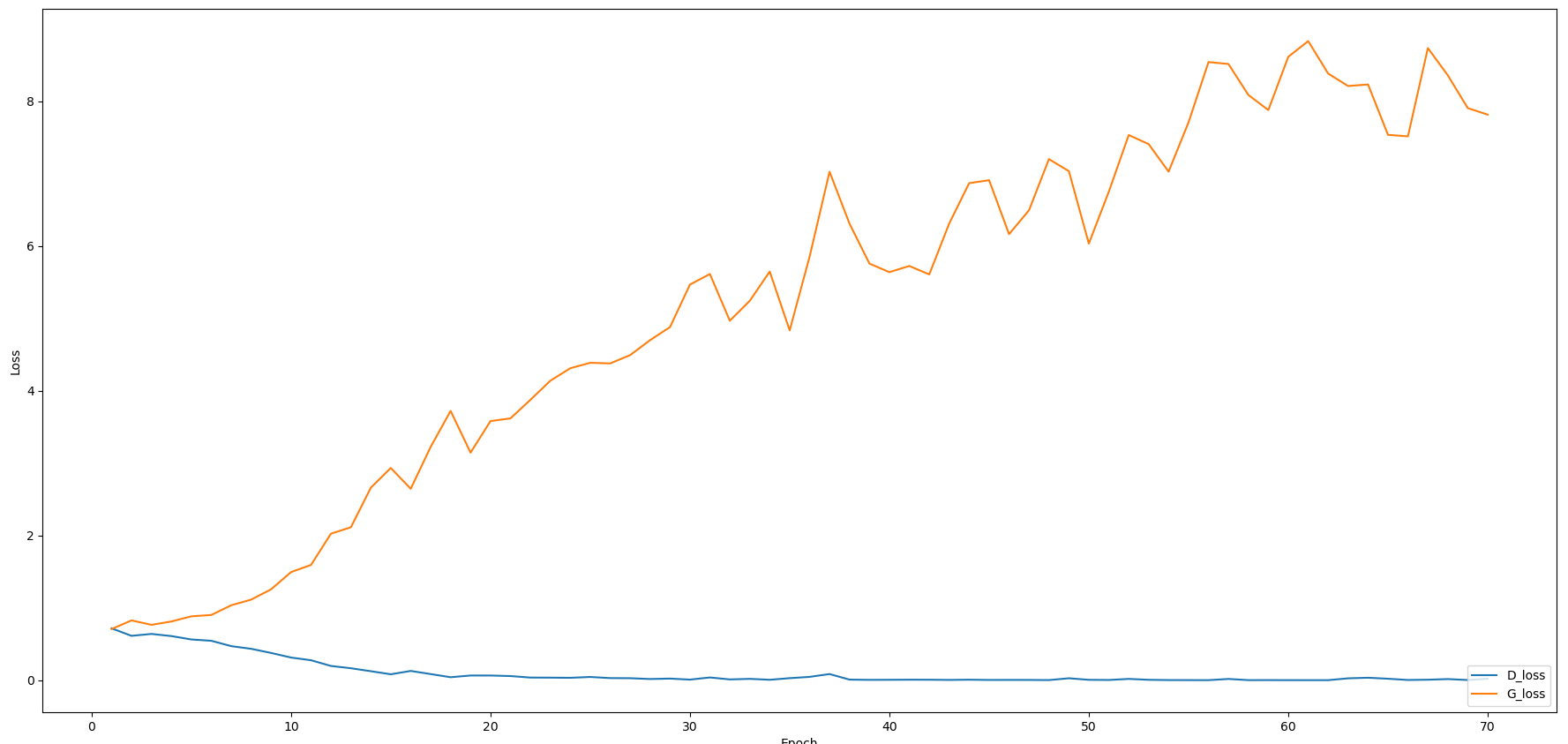
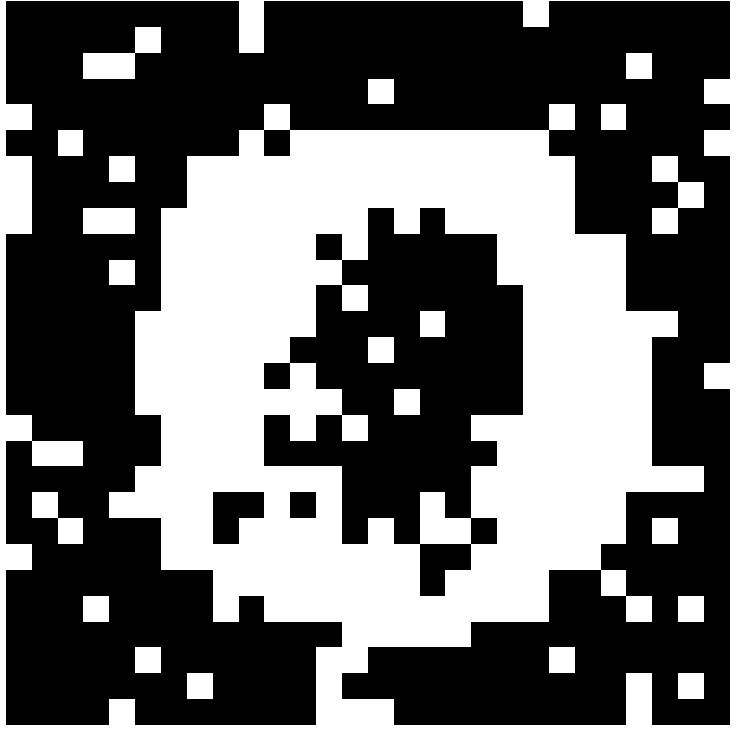
 

1. **调整batchsize（固定epoch=70,lr=0.00005）**

若增加batchsize(batchsize=500)，在没有GPU加速的情况下(4it/s)，运行时间很长，最后的生成效果比原始图片提升不明显，效果如下图所示：



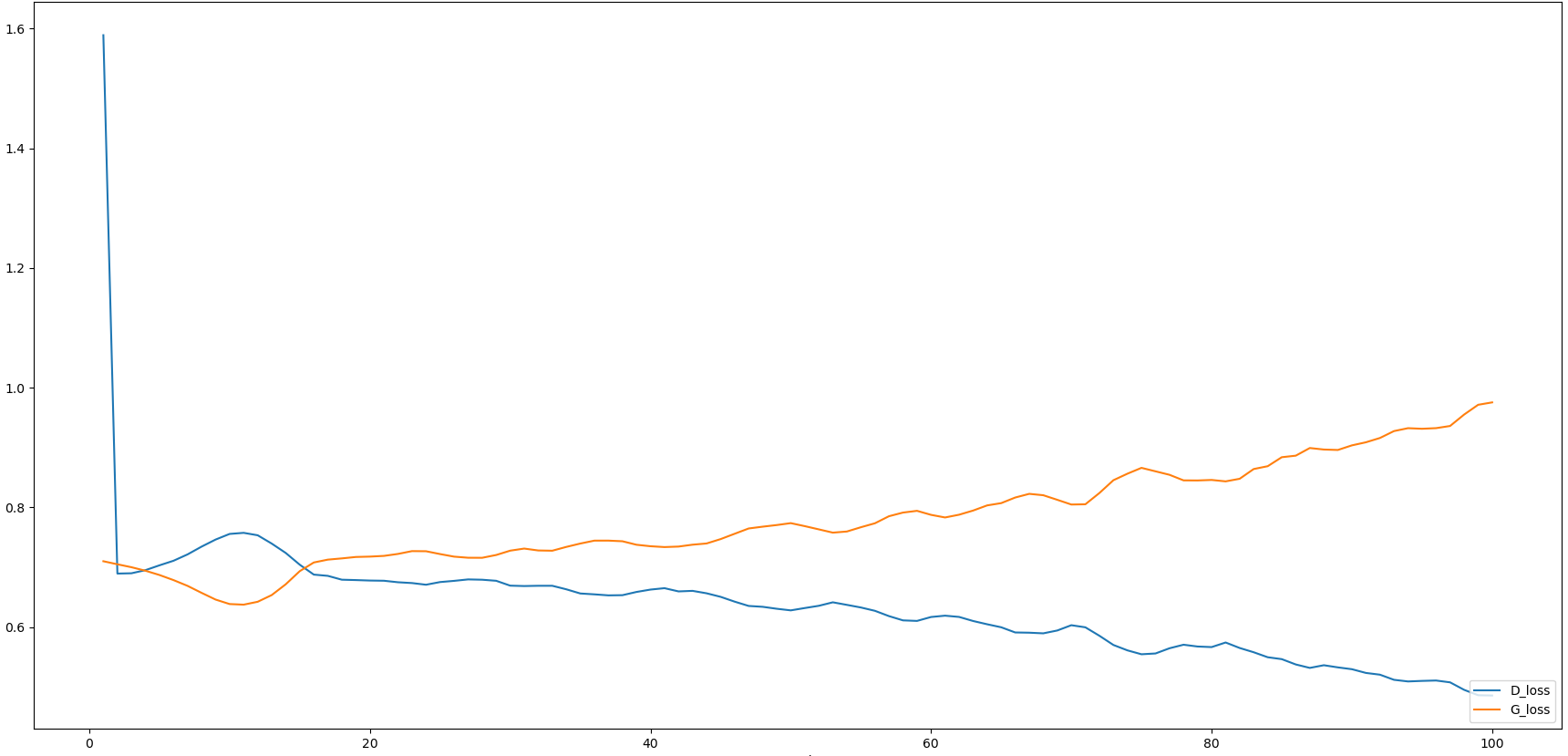
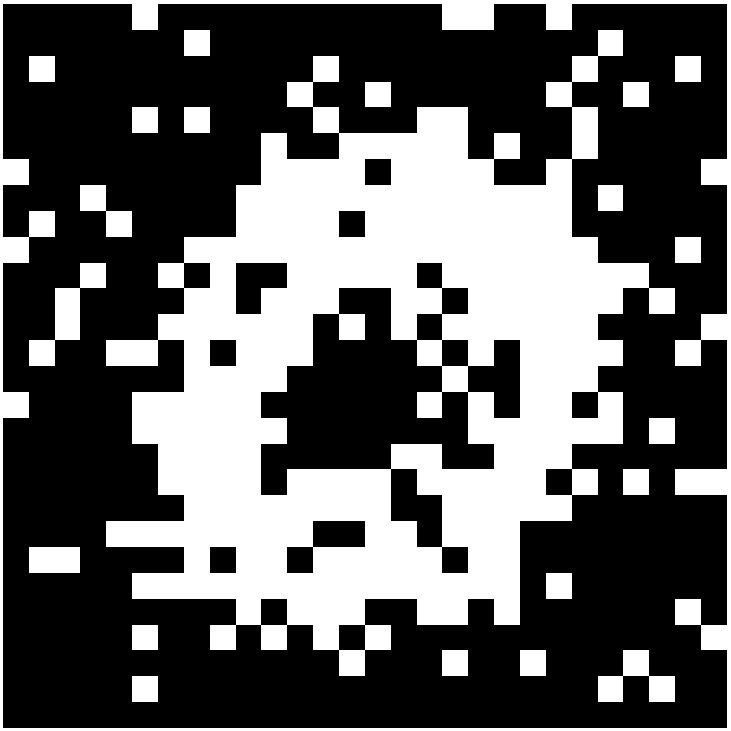
若减小batchsize(batchsize=50)，在没有GPU加速的情况下(4it/s)，运行时间足足有25分钟左右，但最后的生成效果较好，效果如下图所示：



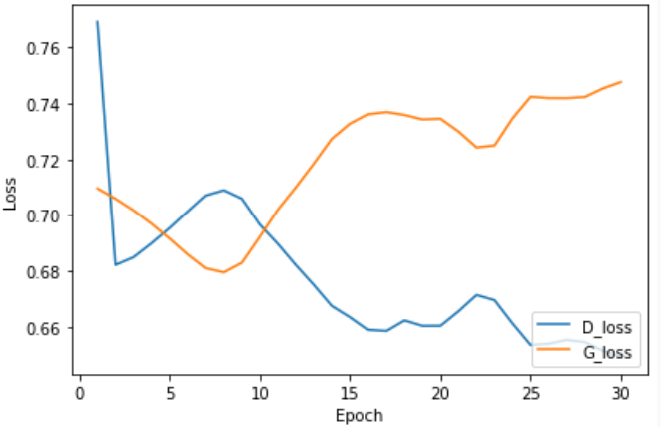
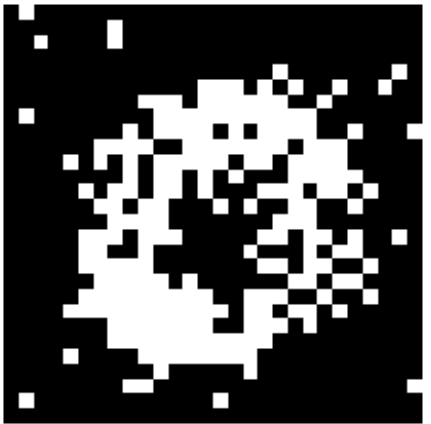
结论：就batch\_size而言，增加batch\_size对最后的生成图片效果几乎是没有帮助的，但若适当减小batch\_size，虽然没有GPU的加速所耗费的时间非常多，但最后生成的效果的客观的。因此最后确定的batchsize=150.

1. **调整inputsize（固定epoch=70,lr=0.00005,batch\_size=150）**

若增加inputsize(inputsize=500)，在没有GPU加速的情况下(4it/s)，运行时间大大增加，生成器的损失逐渐增加，辨别器的损失逐渐减少，最后的生成效果似乎比原始图片有所提升，效果如下图所示：



若减少inputsize(inputsize=50)，最后的生成效果与原始图片没有较大提升，效果如下图所示：



结论：就inputsize而言，在实验结果中观察其对结果的影响较小，就考虑运行时间的因素，我们最终确定其参数为inputsize=200.