摘要

生成对抗网络（GAN）是一类深度生成模型，旨在以无人监督的方式学习目标分布。虽然他们成功应用于许多问题，但训练GAN是一项众所周知的挑战性任务，需要大量的超参数调整，神经网络架构工程以及非常微小的“技巧”。在许多实际应用中的成功加上 缺乏量化GAN失效模式的措施导致了过多的建议损失，正则化和归一化方案，以及网络结构。在这项工作中，我们从实际角度对GAN的当前的问题进行了清晰的观察。我们讨论并评估了常见的陷阱和可重现性问题，在Github上开源我们的代码，并在TensorFlow Hub上提供预训练的模型。

## 引言

深度生成模型是一种强大的（大多数）无监督机器学习模型。 这些模型最近应用于各种应用中，包括图像生成，学习压缩和域适应。

生成对抗网络（GAN）（Goodfellow et al.，2014）是以完全无监督的方式学习这些模型的主要方法之一。GAN框架可被视为双人游戏，其中第一个玩家（生成器），学习将一些简单的输入分布转换为复杂的高维分布（例如自然图像），令第二个玩家（判别器），无法判断样本是从真实分布中采样还是由生成器产生。经典极小极大问题（Goodfellow et al.，2014）的解决方案是纳什均衡，其中任何一方都不能单方面提高。由于生成器和判别器通常被参数化为深度神经网络，因此这个极小极大问题难以解决。

在实践中，使用基于随机梯度的优化方法来训练GAN。除了有训练深度神经网络相关的优化问题之外，GAN的训练还对网络的损失函数的选择，神经网络结构以及使用的正则化和归一化方案非常敏感。 这导致了一系列专注于解决这些问题的研究。(Goodfellow et al., 2014; Salimans et al., 2016; Miyato et al., 2018; Gulrajani et al., 2017; Arjovsky et al., 2017;Mao et al., 2017)

**本文贡献：**本文工作中，我们对这些方法进行了全面的实验分析，并且帮助研究人员和从业人员了解这一领域。我们首先定义了GAN的几个重要设置----损失函数集合，归一化和正则化方案以及最常用的网络体系结构。我们通过超参数优化，在几个流行的大规模数据集上探索这个搜索空间，同时考虑文献中提出的“好”的超参数集，以及通过顺序贝叶斯优化获得的那些超参数集。

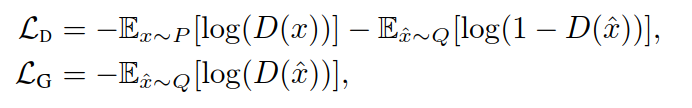
首先，分别测试了各种归一化和正则化方案的效果。 我们证明梯度惩罚（Gulrajani等，2017）以及谱归一化（Miyato等，2018）适用于高容量网络结构。 然后，通过分析损失函数的影响，我们得出结论，非饱和损失（Goodfellow等，2014）在数据集和超参数之间是足够稳定的。最后，表明类似的结论适用于最新模型。 然后，我们讨论一些常见的陷阱，实验可重复性和需要考虑的一些问题。我们提供参考实现包括Github上的训练和评估代码，以及在TensorFlow Hub上提供预训练的模型。

## GAN领域

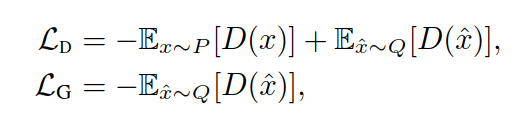
GAN中的主要可以选择的是损失函数，正则化和归一化方法以及网络架构。GAN对这些选择极为敏感。这种情况再加上优化问题和对超参数敏感性使得GAN难以应用于新数据集。下面我们详细介绍主要的选择。

### 损失函数

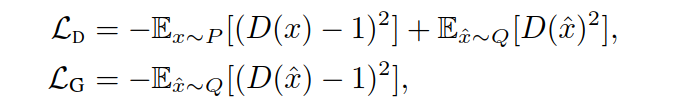
设P表示目标（真实）分布，Q表示模型分布。Goodfellow等人（2014）提出了两个损失函数：minimax GAN和非饱和（NS）GAN。在前者中，判别器最小化二分类任务的负对数似然。 在后者中，生成器最大化生成的样本是真实的概率。 在这里，我们考虑非饱和损失，因为已知它在经验上优于minimax变量。 相应的判别器和生成器损失函数是：



其中D（x）表示从P采样x的概率。在Wasserstein GAN（WGAN）（Arjovsky等，2017）中，作者建议考虑Wasserstein距离而不是Jensen-Shannon（JS）偏差。 相应的损失函数是:



其中判别器输出D（x）∈R并且D需要满足1-Lipschitz。在最优判别器下，相对于最小化生成器损失函数使P和Q之间的Wasserstein距离最小化。关键是确保D的Lipschitzness。最后，我们考虑对应于最小化的最小平方损失（LS）P和Q之间的Pearson χ2差异（Mao et al。，2017）。 相应的损失函数是：



其中D（x）∈R是判别器的输出。直观地说，这种平滑损失函数的饱和速度慢于交叉熵损失。

### 正则化和归一化

**梯度范数惩罚：**这个想法是通过约束其梯度的范数（例如L2范数）来规范判别器D. 在Wasserstein GAN和最佳传输的背景下，该正则化方法很自然地产生，并且在来自P和Q（GP）的样本之间的最佳耦合的点处评估梯度范数（Gulrajani等人，2017）。在GAN训练期间计算该耦合是计算密集的，并且使用这些样本之间的线性插值。梯度范数也可以在数据流形附近受到惩罚，这促使判别器在该区域中是分段线性的。梯度惩罚（GP）正则化方案的缺点在于它在训练期间改变的模型分布Q. 对于Dragan来说，尚不清楚高斯假设的多样性在多大程度上成立。在这两种情况下，计算梯度范数意味着非平凡的运行时间开销。

尽管对特定损失有这些自然的解释，但人们也可以将梯度范数惩罚视为判别器复杂性的经典正则化方法（Fedus等，2018）。 为此，我们还研究了L2正则化对D的影响，这在监督学习中无处不在。

**判别器归一化：**从优化角度（更有效的梯度流，更稳定的优化）以及从表示的角度来看，归一化判别器可能是有用的 - 神经网络中层的表示丰富度取决于关于相应权重矩阵的谱结构（Miyato等，2018）。

从优化的角度来看，通常应用于深度神经网络训练的几种归一化技术已应用于GAN，即批量归一化（BN）（Ioffe＆Szegedy，2015）和层归一化（LN）（Ba等，2016）。 前者在Denton等人的研究中进行了探索。（2015年）并由Radford等人进一步推广。（2016），而后者在Gulrajani等人进行了研究（2017年）。 这些技术用于跨批次(BN) 或跨特征(LN)标准化激活，观察到这两种激活都改善了经验性能。

从表示的角度来看，可以将神经网络看作是(可能是非线性的)映射的组合，并分析它们的光谱特性。特别地，对于一个有界算子，只要控制每个映射的算子范数就足够了。Miyato等人(2018)采用了这种方法，他们建议将每个权值矩阵(包括表示卷积内核的矩阵)除以它们的频谱范数。分析了谱归一化的结果表明，相对于其他归一化方法，谱归一化的判别器具有更高的秩。

### 生成器和判别器的结构

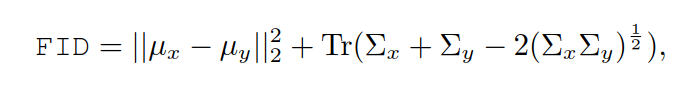
我们在本文研究中探索了两类架构:deep convolutional generation adversarial networks (DC-GAN) (Radford et al.， 2016)和residual networks (ResNet) (He et al.， 2016)，这两类架构在GAN的研究中是普遍存在的。最近，Miyato等人(2018)定义了一种DCGAN变体，即SNDCGAN。除了小的更新(参见第4节)，与DCGAN的主要区别是使用了8层判别器网络。表4总结了这两个网络的详细信息。另一种架构是ResNet19，它在生成器中有5个ResNet块，在判别器中有6个ResNet块，可以对128×128张图像进行操作。我们遵循Miyato等人(2018)的ResNet设置，只是我们简化了判别器的设计。网络结构细节汇总在表5a和表5b中。通过这种设置，我们能够重现Miyato等人(2018)的研究结果。关于各种ResNet修改的消融研究见附录。

### 评估准则

我们专注于几个最近提出的非常适合图像领域的指标。有关量化指标的深入概述，我们将读者推荐给Borji（2019）。

**Inception Score（IS）：**Salimans等人(2016)提出的IS提供了一种定量评估生成样本质量的方法。从直观上看，包含有意义对象的样本的条件标签分布应该是低熵的，样本的可变性应该是高的。这可以表示为= exp(Ex∼Q[dKL(p(y | x)， p(y))])。作者发现这个分数与人类注释者的分数有很好的相关性。缺点包括对标签上的先验分布不敏感以及没有适当的距离。

**FID：**在Heusel等人(2017)提出的这种方法中，来自P和Q的样本首先嵌入到特征空间(InceptionNet的特定层)中。然后，假设嵌入数据服从多元高斯分布，估计均值和协方差。最后计算了这两个高斯函数之间的Frechet距离，即



其中（μx，Σx）和（μy，Σy）分别是来自P和Q的嵌入样本的均值和协方差。作者认为，FID与人类的判断相一致，并且比IS更加稳健。 此外，分数对生成的样本的视觉质量敏感，在生成的样本中引入噪声或伪影将减少FID。与IS相比，FID可以检测到类内模式丢弃，每个类只生成一个图像的模型将具有良好的IS，但是具有错误的FID（Lucic等，2018）。

**Kernel Inception Distance (KID)：**Bi’nkowski等人(2018)认为FID没有无偏估计量，建议将KID作为无偏估计量。在附录B中，我们对KID和FID进行了实验比较，发现这两个指标都有很强的相关性(LSUN-BEDROOM的Spearman秩级相关系数为0.994,CELEBA-HQ-128数据集的相关系数为0.995)。因此，我们关注FID，因为它很可能导致相同的排名。

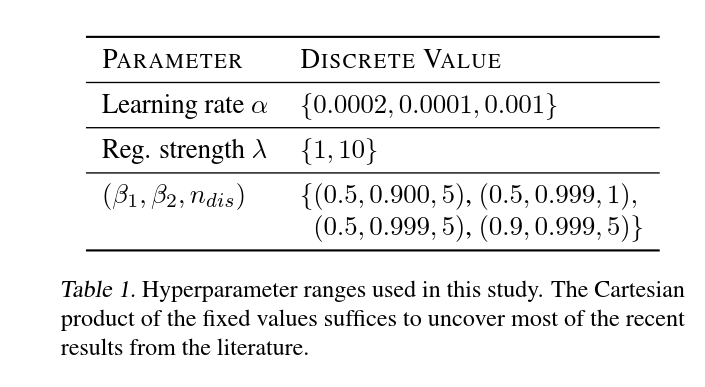
### 数据集

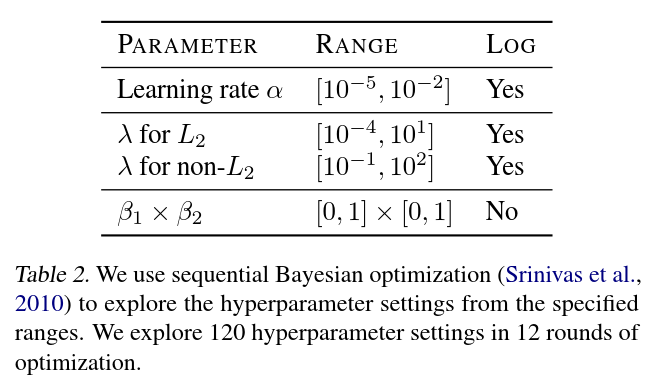
我们考虑了三个数据集，即CIFAR10、CELEBA-HQ-128和LSUN-BEDROOM。LSUN-BEDROOM数据集包含略多于300万张图像(Yu et al.，2015)。将图像随机划分为训练集和测试集，其中使用30588张图像作为测试集。其次，使用CELEBA-HQ数据集，包含30K张图像(Karras et al.， 2018)。我们使用作者提供的代码得到的128×128×3版本。我们使用3K个例子作为测试集，其余的图像作为训练集。最后，我们还使用CIFAR10数据集，其中包含70K张图像(32×32×3)，划分为60K个训练图像和10K个测试图像。FID的基线分数为12.6分(CELEBA-HQ-128)， 3.8分(sun - bedroom)， 5.19分(CIFAR10)。FID计算的细节在第4节中给出。

### Exploring the GAN Landscape

GANs的搜索空间非常大:探索所有损失、归一化和正则化方案的所有组合，以及网络结构，这些都超出了实际领域。相反，在本研究中，我们为每个数据集分析这个搜索空间的几个部分。特别是，为了确保我们可以重现现有的结果，我们在CIFAR10上对这个搜索空间的子集执行了一个研究。然后我们分析了这些模型在CELEBA-HQ-128和LSUN-BEDROOM中的性能。在3.1节中，我们修改了除正则化和归一化方案之外的所有内容。在第3.2节中，我们修改了除损失之外的所有内容。最后，在3.3节中，我们修改了除网络结构之外的所有内容。这使我们能够解耦这些选择，并提供一些在实践中最重要的见解。

正如Lucic等人(2018)指出的，阻碍进一步进展的一个主要问题是超参数调优——目前，社区已经收集到一小组参数值，这些值在一些数据集上工作，而在其他数据集上可能完全失败。在本研究中，我们结合文献中发现的最佳超参数设置(Miyato et al.， 2018)，并执行顺序贝叶斯优化(Srinivas et al.，2010)，以可能发现更好的超参数设置。简而言之，在序列贝叶斯优化中，首先要评估一组超参数设置(可能是随机选择的)。然后，根据这些超参数获得的分数，选择下一组超参数组合，以平衡探索(寻找可能执行得很好的新超参数设置)和利用(选择接近最佳执行设置的设置)。然后，我们考虑性能最好的模型，并讨论计算开销的影响。

我们总结了表1中固定的超参数设置，其中包含了近期发表的“好”参数(Fedus et al.， 2018;Miyato et al ., 2018;Gulrajani等，2017)。特别地，我们考虑这些参数的笛卡尔积来获得24个超参数设置来减少生存偏差。最后，为了提供一个公平的比较，我们对表2中提供的参数范围执行顺序贝叶斯优化(Srinivas et al.， 2010)。我们跑了12轮。与oracle进行了12次通信)，每一次都根据之前迭代结果的FID分数选择了一批10个超参数集。当我们研究每个生成器更新中判别器更新的数量(1或5)时，这将导致额外的240个超参数设置，在某些情况下，这些超参数设置的性能优于以前已知的超参数设置。所有实验的批大小都设置为64。对于LSUN-BEDROOM数据集和CELEBA-HQ-128数据集，我们使用固定数量的识别器更新步骤为100K，对于CIFAR10数据集，我们使用200K。我们应用Adam优化器(Kingma & Ba, 2015)。



## 实验结果及讨论

由于每个数据集有4个主要组件(loss、architecture、regularization、normalization)需要分析，因此无法对整个空间进行探索。因此，我们选择了一个更实际的解决方案——我们保持一些维度不变，而改变其他维度。我们强调两个方面:

1. 我们使用各种超参数设置对模型进行训练，这些超参数设置既有预先定义的，也有通过顺序贝叶斯优化得到的。然后我们计算了前5%训练模型的FID分布。FID中值越低，模型越好。从优化的角度来看，方差越小，模型越稳定。
2. 根据FID计算预算（用于训练）和模型质量之间的权衡。 直观地说，考虑到有限的计算预算（只能训练k个不同的模型），应该选择哪种模型？ 显然，在实践中应该优选使用相同计算预算实现更好性能的模型。为了计算固定预算k最小可达到的FID，我们模拟了一个从业者试图为他们的模型找到一个好的超参数设置：我们将预算的一部分用于最近研究中报告的“好”超参数设置，然后探索新的设置 （即使用贝叶斯优化）。 由于这是一个随机过程，我们重复1000次并记录可达到的最小FID的平均值。

由于训练对初始权重敏感，我们训练模型5次，每次使用不同的随机初始化，并报告中位数FID。 通过顺序贝叶斯优化获得的模型的FID方差，由exploration-exploitation strategy隐式处理。

### 正则化归一化

本研究的目的是比较文献中各种正则化和归一化方法的相对性能，即批量归一化(BN) (Ioffe & Szegedy, 2015)、层归一化(LN) (Ba et al.， 2016)、谱归一化(SN)、梯度惩罚(GP) (Gulrajani et al.， 2017)、Dragan惩罚(DR) (Kodaliet al.， 2017)和L2正则化。我们将损失修改为非饱和损失(Goodfellow et al.， 2014)和带有表5a中描述的生成器和判别器架构的ResNet19。我们分析了损失函数在3.2节和架构在3.3节中的影响。我们考虑了CELEBA-HQ-128和LSUN-BEDROOM的超参数设置，如表1和表2所示。

结果如图1所示。我们注意到，向判别器添加批量归一化会降低性能。其次，梯度惩罚可以帮助，但它不能稳定训练。事实上，在损失和正则化强度之间取得平衡并非易事。谱归一化有助于提高模型质量，比梯度惩罚具有更高的计算效率。这与Zhang等人(2019)的最新研究结果一致。与损失研究类似，使用GP惩罚的模型可能受益于判别器与生成器更新的5:1比例。此外，在一项单独的消融研究中，我们观察到，运行额外的10万步优化程序可能会提高GP惩罚模型的性能。

### 损失函数的影响

在这里，我们研究上述发现是否也适用于损失函数的变化。除了非饱和损失(NS)，我们还考虑了最小二乘损失(LS) (Mao et al.， 2017)或沃瑟斯坦损失(Wasserstein loss, WGAN) (Arjovsky et al.， 2017)。我们使用带有ResNet19的生成器和判别器架构，详见表5a。我们考虑了最突出的归一化和正则化方法:梯度惩罚(Gulrajani et al.， 2017)和谱归一化(Miyato et al.， 2018)。其他参数详见表1。我们还对最近流行的hinge loss进行了研究(Lim & Ye, 2017;Miyato等，2018;Brock等人，2019)，并将其呈现在附录中。

结果如图2所示。谱归一化提高了两个数据集的模型质量。类似地，梯度惩罚也会有所帮助，但是找到一个好的正则化折衷并不简单，而且需要大量的计算开销。使用GP惩罚的模型受益于识别器与生成器更新的比例为5:1 (Gulrajani et al.， 2017)。

### 网络结构的影响

一个有趣的实际问题是，我们的发现是否也适用于不同的网络结构。为此，我们还对Miyato等人(2018)的SNDCGAN进行了研究。考虑不饱和GAN损失、梯度惩罚和谱归一化。虽然对于较小的架构来说，正则化并不是必需的(Lucic et al.， 2018)，但是由于更深层次的架构和优化考虑，正则化和归一化效果可能变得更加相关。结果如图3所示。我们注意到，这两种体系结构实现了类似的结果，并从正则化和归一化中获益。对于这两种体系结构，谱归一化的性能都比baseline好。

**正则化和归一化：**一个常见的观察结果是，判别器的Lipschitz常数对模型的性能至关重要，可以预见同时进行正则化和归一化可以提高模型的质量。为了量化这种效果，我们将损失固定为非饱和损失(Goodfellow et al.， 2014)，使用Resnet19体系结构(如上所述)，结合几种标准化和正则化方案，使用表1中所示的超参数设置和随机选择的24个参数。结果如图4所示。我们注意到，可以从额外的正则化和归一化中获益。然而，在FID中需要投入大量的计算工作来获得一些边际收益。然而，在有足够的计算资源的情况下，我们提倡同时进行正则化和归一化——谱归一化和层归一化在实践中似乎效果很好。

## 研究的挑战

在本节中，我们将重点放在尝试重现现有结果并提供公平准确比较时遇到的几个陷阱。

**准则：**FID分数的计算方式似乎已经存在差异：（1）一些作者报告了训练数据的分数，产生了50K训练和50K生成样本之间的FID（Unterthiner等，2018）。 有些人选择报告基于10K测试样本和5K生成样本的FID并使用自定义实现（Miy-ato等，2018）。 最后，Lucic等人。 （2018）报告关于测试数据的分数，特别是10K测试样本和10K生成样本之间的FID。 细微的差异将导致报告的FID之间的不匹配，在某些情况下超过10％。 我们认为应该根据测试数据集计算FID。此外，只要有可能，就应该使用与先前报告的结果相同数量的实例。 为此，我们在CIFAR10和LSUN-BEDROOM上使用10K测试样品和10K生成的样品，在Lucic等人（2018）中使用CELEBA-HQ-128上的3K和3K。

**网络结构细节：**即使在流行的体系结构中，比如ResNet，仍然需要做出许多设计决策，这些决策通常在论文的结果中被省略。这些包括ResNet块的精确设计(层的顺序、何时应用ReLu、何时上采样和下采样、使用多少过滤器)。其中一些差异可能会导致潜在的不公平比较。因此，我们建议使用本工作中呈现的体系结构作为坚实的基线。关于各种ResNet修饰的消融研究见附录。

**数据集：**常见问题与数据集处理有关 - LSUN-BEDROOM是否始终对应于同一数据集？ 在大多数情况下，用于放大或裁剪的精确算法不明确，这导致“相同”数据集上的结果之间的不一致。

**实施细则和非确定性：**一个主要问题是论文中提出的算法与在线提供的代码之间的不匹配。 我们意识到良好的实现与给定模型的不良实现之间存在着令人尴尬的巨大差距。 因此，当没有代码可用时，人们不得不猜测是做了哪些修改。 另一个特别棘手的问题是从训练过程中消除随机性。在确定数据排序和初始权重之后，通过训练相同模型两次获得相同的分数是非常重要的，因为某些GPU操作中存在随机性（Chetlur等，2014）。 禁用导致非确定性的优化通常会导致运行时间损失。

虽然这些单独考虑的问题似乎都是次要的，但它们组合在一起形成了一层薄雾，在实际应用和研究过程中引入了摩擦。

## 相关工作

Lucic等人(2018)最近对GANs和变分自编码器进行了大规模研究。作者考虑了几种损失函数和正则化器，研究了损失函数对FID分数的影响，使用了低到中等复杂度的数据集(MNIST、CIFAR10、CELEBA)和单一的神经网络结构。在这个有限的设置下，作者发现，最近引入的模型和原来的非饱和GAN之间没有统计学上的显著差异。Fedus等人(2018)最近发表了一项关于梯度-范数正则化在GANs中的影响的研究。作者认为梯度惩罚也适用于非饱和GAN，在一定程度上降低了对超参数选择的敏感性。在最近的光谱归一化研究中，作者对相互竞争的正则化和归一化方法进行了小规模研究(Miyato et al.， 2018)。我们很高兴地报告，我们可以复制这些结果，并在附录中给出。

受这些作品的启发，并以Lucic等人(2018)的可用开放源代码为基础，我们在其中考虑的所有维度上都采取了一个额外的步骤:更复杂的神经体系结构、更复杂的数据集以及更复杂的正则化和归一化方案。

## 总结和展望

本文研究了归一化和归一化方案对GAN训练的影响。我们考虑最先进的方法，并改变损失函数和网络结构。我们研究了这些设计和选择对生成的样本质量的影响，我们通过最近引入的定量度量来评估这些样本的质量。

我们公平且全面的经验评估表明，当计算预算有限时，应将非饱和GAN损失和谱归一化视为将GAN应用于新数据集时的默认选择。 给定额外的计算预算，我们建议添加Gulrajani等人的梯度惩罚。并训练模型直到收敛。 此外，我们观察到两类流行的神经架构在所考虑的数据集中都能表现良好。 一项单独的消融研究发现，在ResNet风格的体系结构中应用的大多数变化，会导致样本质量的微小改进。

作为这项大规模研究的结果，我们发现了阻碍准确和公平比较的常见陷阱，并提出了具体的行动，以揭开未来结果的神秘面纱，其中度量标准、数据集预处理、非决定论和缺失的实现细节尤其引人注目。我们希望这项工作，连同开源实现和经过训练的模型，将成为未来GAN研究的坚实基础。

未来的工作应该仔细评估需要进行大规模训练的模型，如BigGAN（Brock等,2019），具有定制架构的模型（Chen等,2019; Karras等,2019; Zhang等,2019） 最近提出的正则化技术（Roth et al,2017; Mescheder et al.,2018），以及其他稳定训练的建议（Chen et al.，2018）。 此外，考虑到CGAN的流行，人们应该探索这些见解是否转移到条件设置。 最后，鉴于FID和IS的缺点，使用最近提出的指标进行额外的定量评估可以带来新的见解（Sajjadi等,2018; Kynkaanniemi等,2019）。

### 致谢

感谢Michael Tschannen对本文的详细评论。

### ResNet消融研究

我们注意到与来自github的实现相比，Resnet架构有六个细微差别。我们进行了消融研究以验证这些差异的影响。 图7显示了消融研究的影响，详细描述如下。

1. 默认情况：ResNet CIFAR架构，具有谱归一化和非饱和GAN损失
2. 残差连接：使用输入作为判别器ResBlock中残差连接的输出。 默认情况下，它是一个带有3x3内核的卷积层。
3. CIN: 使用ci作为判别器ResBlock隐藏层输出通道数目。默认情况下设置为C\_0, 但是Miyatoet al. (2018)使用c\_o作为第一层的输出通道数，其余的通道数设置为c\_i
4. OPT: 判别器的第一个残差块儿设置一些优化，包括：（1）没有Relu激活（2）一个包含残差连接的卷积层（3）在残差块儿内部，用c\_o替代c\_i
5. CIN OPT： 将CIN和OPT的设置结合。
6. SUM: 判别器输出层使用reduce\_sum操作。默认使用reduce\_mean
7. TAN: 使用Tanh作为生成器输出层激活，判别器输入值的范围为[-1, 1]。默认使用sigmoid激活，判别器输入范围为[0, 1]
8. EPS: 生成器中BN层使用较大的epsilon：2e-5。TensorFlow默认为1e-5。
9. ALL：将上述的设置综合使用

残差块的设计图

FID指标的介绍