分析和改善 StyleGAN 的图像质量

Tero Karras NVIDIA
Samuli Laine NVIDIA
Miika Aittala NVIDIA
Janne Hellsten NVIDIA
Jaakko Lehtinen NVIDIA and Aalto University
Timo Aila NVIDIA

摘 要

基于样式的 GAN 架构（StyleGAN）在数据驱动的无条件生成图像建模中达到了最先进的结果。我们将揭露和分析其一些特征伪像的出现原因，并提出模型架构和训练方法方面的改进以解决这些问题。特别是，我们重新设计了生成器归一化方法，重新审视了渐进式增长架构，并对生成器施加了正则化，以在从潜在矢量到图像的映射中鼓励更良好的质量。除了改善图像质量外，使用路径长度调节器还带来了额外的好处，即生成器变得非常容易反转。这使得可以可靠地检测图像是否由特定网络生成。我们进一步可视化生成器如何充分利用其输出分辨率，并确定网络容量问题，从而激励我们训练更大的模型以进一步提高质量。总体而言，我们改进的模型在现有的分布式指标质量和感知的图像质量方面都重新定义了无条件图像建模的最先进技术指标。

1. 介绍

通过生成方法，尤其是生成对抗网络（GAN）[15]生成的图像的分辨率和质量正在迅速提高[23，31，5]。目前，用于高分辨率图像合成的最新方法是 StyleGAN [24]，它已被证明可以在各种数据集上可靠地工作。我们的工作重点是修复其特征伪像，并进一步提高结果质量。

StyleGAN [24]的显著特征是其非常规的生成器体系结构。映射网络 f 不再将输入的潜在编码$\mathbf{z} \in \mathcal{Z}$仅馈送到网络的开头，而是将其转换为中间的潜在编码$\mathbf{w} \in \mathcal{W}$。然后仿射变换生成控制图层的样式，并通过自适应实例归一化（AdaIN）[20、9、12、8]参与合成网络 g 进行合成。 另外，通过向合成网络提供额外的随机噪声图来促进随机变化。

[24，38]已经证明，这种设计允许中间潜空间 W 的纠缠比输入潜空间 Z 的纠缠小得多。在本文中，我们所有的分析都只集中在 W 上，因为它是从合成网络相关的潜码空间角度来看问题。

许多观察者已经注意到由 StyleGAN [3]生成的图像中的特征伪影。我们确定了造成这些伪影的两个原因，并描述了消除这些伪影的体系结构和改进的训练方法。首先，我们研究了常见的斑点状伪像的起源，并发现生成器创建它们是为了规避其体系结构中的设计缺陷。在第 2 节中，我们重新设计了生成器中使用的归一化，该归一化消除了伪像。其次，我们分析了与渐进式增长相关的伪影[23]，该伪影在稳定高分辨率 GAN 训练方面非常成功。我们提出了一个可实现相同目标的替代设计-即训练从关注低分辨率图像开始，然后逐渐将焦点转移到越来越高的分辨率-而不在训练期间更改网络拓扑结构。这种新设计还使我们能够合理地理解所生成图像的有效分辨率，该分辨率实际上低于预期值，从而导致了网络容量的增加（第 4 节）。

使用基于生成方法产生的图像质量的定量分析仍然是一个具有挑战性的话题。Frechet 初始距离（FID）[19]测量了 InceptionV3 分类器的高维特征空间中两个分布的密度差异[39]。精确度和召回率（P&R）[36，27]通过显式量化分别类似于训练数据的生成图像的百分比和可以生成的训练数据的百分比，从而提供了额外的鉴定依据。我们使用这些指标来量化改进方法。

FID 和 P&R 均基于分类器网络，最近已证明该分类器网络侧重于纹理而不是形状[11]，因此，这些度量不能准确地捕获图像质量的所有方面。我们观察到感知路径长度（PPL）度量[24]被引入作为一种估计潜在空间插值质量的方法，

图 1.实例归一化导致 StyleGAN 图像中出现类似水滴的伪影。这些在生成的图像中并不总是很明显，但是如果我们查看生成器网络内部的激活层，则问题始终存在，在所有从 64x64 分辨率开始的特征图中。这是困扰所有 StyleGAN 图像的系统性问题。

与形状的一致性和稳定性相关。在此基础上，我们对合成网络进行了正则化处理，以支持平滑映射（第 3 节），并明显实现了质量的提高。为了抵消其计算代价，我们还建议不那么频繁地执行所有正则化，并观察到这样做可以在不影响效果的情况下进行。

最后，我们发现，使用新的路径长度正则化生成器比使用原始 StyleGAN，将图像投影到潜在空间 W 的效果明显更好。这具有实际意义，因为它使我们能够可靠地判断是否使用特定的生成器生成了给定的图像（第 5 节）。

我们的实施和经过训练的模型可在下方地址获得：
https://github.com/NVlabs/stylegan2

2. 消除因归一化导致的伪像

我们首先观察到 StyleGAN 生成的大多数图像都呈现出类似于水滴的特征性斑点状伪像。如图 1 所示，即使液滴在最终图像中可能不明显，它也会出现在生成器的中间特征图中（见底部 1）。异常开始出现在 64×64 分辨率附近，并出现在所有特征图中，并且在更高的分辨率下变得越来越强。这种持续出现的伪像令人困惑，因为判别器应该能够检测到它。

我们将问题精确定位到 AdaIN 运算中，该运算可分别归一化每个特征图的均值和方差，从而潜在地破坏在特征量级中相对于彼此发现的任何信息。我们假设液滴伪像是生成器故意将信号强度信息偷偷经过实例归一化的结果：通过创建主导统计数据的强大的局部尖峰，生成器可以像其他地方一样有效缩放信号。我们的假设受到以下发现的支持：当从生成器中删除归一化步骤时，如下所述，液滴伪像会完全消失。

[1] 在极少数情况下（可能占图像的 0.1%），液滴会丢失，从而导致图像严重损坏。有关详细信息，请参见附录 A。

2.1. 生成器架构修正

我们将首先修改 StyleGAN 生成器的几个细节，以更好地促进我们重新设计归一化。就质量指标而言，这些变化本身产生中性或小小的积极影响

图 2a 显示了原始的 StyleGAN 合成网络 g [24]，在图 2b 中，我们通过显示权重和偏差并将 AdaIN 操作分解为其两个组成部分：归一化和调制，将特征图扩展为完整细节。这使我们可以重新绘制概念上的灰色框，以便每个框都指示网络中激活一种样式的部分（即“样式块”）。有趣的是，原始的 StyleGAN 在样式块内施加了偏置和噪音，使它们的相对影响与当前样式的大小成反比。我们观察到，通过将这些操作移到样式块之外（它们在未标准化的数据上进行操作），可以获得更可预测的结果。此外，我们注意到，在此更改之后，仅对标准偏差进行标准化和调制就足够了（即，不需要均值）。偏置，噪声和归一化对恒定输入的应用也可以安全地消除，而没有明显的缺点。此变体如图 2c 所示，并作为我们重新设计的归一化的起点。

2.2. 实例归一化修正

鉴于实例归一化似乎过于强大，我们如何在保留样式特定比例的效果的同时松弛它呢？我们排除了批量归一化 [21]，因为它与高分辨率合成所需的小型小批量不兼容。或者，我们可以简单地删除归一化。尽管实际上稍微改善了 FID [27]，但这使样式的效果得以累积而不是特定于比例，从而实质上失去了 StyleGAN 提供的可控制性（请参见视频）。现在，我们将提出一种替代方法，该方法在保留可控制性的同时删除了伪像。主要思想是基于传入特征图的预期统计量进行归一化，但没有显式强制。

回想一下，图 2c 中的样式块由调制，卷积和归一化组成。让我们开始考虑

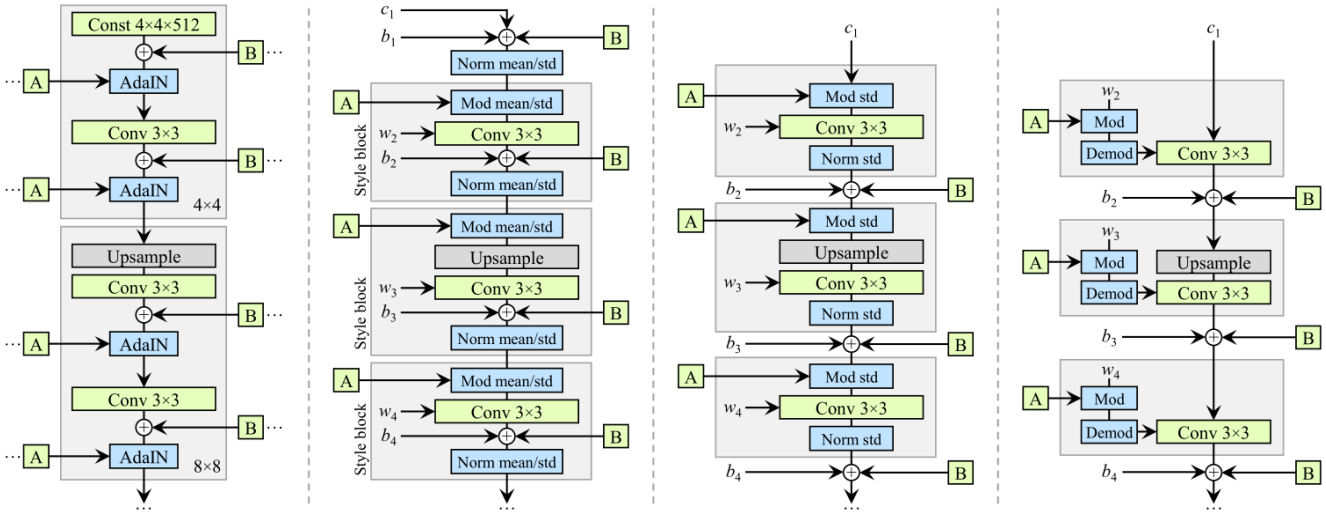


图 2.我们重新设计了 StyleGAN 合成网络的架构。（a）原始 StyleGAN，其中 A 表示从 W 学习的仿射变换，产生样式向量，而 B 表示噪声广播操作。（b）完整细节相同的图。在这里，我们将 AdaIN 分解为显式归一化后再进行调制，然后对每个特征图的均值和标准差进行操作。我们还对学习的权重（w），偏差（b）和常数输入（c）进行了注释，并重新绘制了灰色框，以便每个框都激活一种样式。激活函数（Leaky ReLU）总是在添加偏置后立即应用。（c）我们对原始架构进行了一些改动，这些改动在正文中是有效的。我们从一开始就删除了一些多余的操作，将 b 和 B 的相加移到样式的有效区域之外，并仅调整每个要素图的标准偏差。（d）修改后的体系结构使我们能够用“解调”操作代替实例归一化，该操作适用于与每个卷积层相关的权重。

卷积后的调制效果。调制根据输入样式缩放卷积的每个输入特征图，也可以通过缩放卷积权重来实现：

$$w'_{ijk} = s_i \cdot w_{ijk}, \tag{1}$$

其中 w 和w'分别是原始权重和调制权重，s_i是与第 i 个输入特征图相对应的比例，而 j 和 k 分别列举输出特征图和卷积的空间下标。

现在，实例归一化的目的是从卷积输出特征图的统计信息中实质上消除 s 的影响。我们观察到可以更直接地实现这一目标。让我们假设输入激活是独立同分布带有单位标准偏差的随机变量。经过调制和卷积后，输出激活的标准偏差为

$$\sigma_j = \sqrt{\sum_{i,k} {w'_{ijk}}^2}, \tag{2}$$

即，通过相应权重的 L2 范数来缩放输出。随后的标准化旨在将输出恢复为单位标准偏差。基于等式 2，如果我们将每个输出特征图 j 缩放（“解调”）$1/\sigma_j$，则可以实现此目的。 或者，我们可以

再次将其嵌入到卷积权重中：

$$w''_{ijk} = w'_{ijk} \Big/ \sqrt{\sum_{i,k} {w'_{ijk}}^2 + \epsilon}, \tag{3}$$

其中ϵ是避免数值问题的小常数。

现在，我们已经将整个样式块嵌入到单个卷积层，其权重使用公式 1 和 3 基于 s 进行调整（图 2d）。与实例归一化相比，我们的解调技术较弱，因为它基于关于信号的统计假设，而不是特征图的实际内容。类似的统计分析已在现代网络初始化程序中广泛使用[13，18]，但我们不知道它先前已被用来替代依赖数据的归一化。我们的解调也与权重归一化[37]有关，权重归一化[37]执行与重新设定权重张量相同的计算。先前的工作已经证明权重归一化在 GAN 训练中是有益的[42]。

我们的新设计消除了特征伪像（图 3），同时保持了完全的可控制性，如随附视频所示。FID 基本上不受影响（表 1，行 A，B），但是从 Precision 到 Recall 有着明显的转变。我们认为这通常是合乎需要的，因为可以通过截断将召回率转换为精度，但事实并非如此[27]。实际上，我们的设计可以是

Configuration	FFHQ, 1024×1024				LSUN Car, 512×384			
	FID	Path length	Precision	Recall	FID	Path length	Precision	Recall
A Baseline StyleGAN [24]	4.40	195.9	**0.721**	0.399	3.27	1484.5	**0.701**	0.435
B + Weight demodulation	4.39	173.8	0.702	0.425	3.04	862.4	0.685	0.488
C + Lazy regularization	4.38	167.2	0.719	0.427	2.83	981.6	0.688	0.493
D + Path length regularization	4.34	139.2	0.715	0.418	3.43	651.2	0.697	0.452
E + No growing, new G & D arch.	3.31	**116.7**	0.705	0.449	3.19	471.2	0.690	0.454
F + Large networks	**2.84**	129.4	0.689	**0.492**	**2.32**	**415.5**	0.678	**0.514**

表 1.主要结果。对于每次训练，我们选择 FID 最低的训练快照。我们使用不同的随机种子计算每个指标 10 次，并报告了它们的平均值。“路径长度”列对应于基于 W [24]中的路径端点计算的 PPL 度量。对于 LSUN 数据集，我们报告的路径长度不包含最初为 FFHQ 建议的中心作物。 FFHQ 数据集包含 70k 图像，我们在训练过程中显示了 25M 的判别器图像。对于 LSUN CAR，相应的数字是 893k 和 57M。

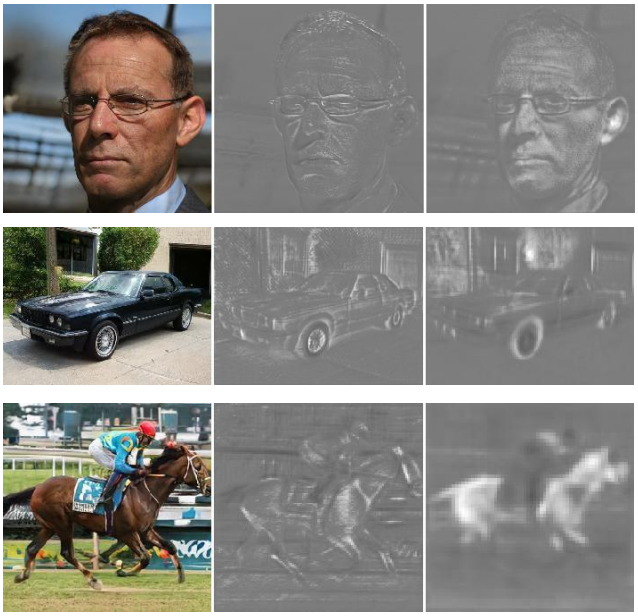
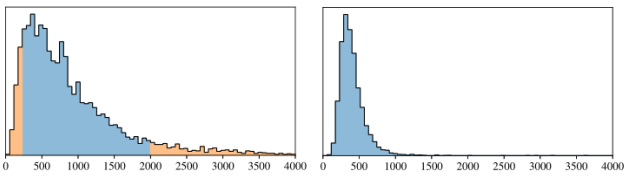


图 3.用解调代替归一化可从图像和激活中删除特征伪像。

(a) Low PPL scores (b) High PPL scores

图 4.使用基线 StyleGAN（表 1 中的配置 A）在感知路径长度和图像质量之间建立联系。（a）PPL 低（≤第 10 个百分点）的随机例子。（b）PPL 高（≥第 90 个百分点）的示例。 PPL 分数与图像的语义一致性之间存在明显的相关性。



(a) Baseline StyleGAN (b) Our method (config F)

图 5.（a）使用基线 StyleGAN（表 1 中的配置 A，FID = 8.53，PPL = 924）生成的单个图像的 PPL 分数分布。对应于图 4 的百分比范围以橙色突出显示。（b）我们的方法（配置 F）显著改善了 PPL 分布（显示了具有相同 FID = 8.53，PPL = 387 的快照）。

如附录 B 所述，使用分组卷积有效地实现该功能。为避免必须在公式 3 中考虑激活函数，我们对激活函数进行缩放，以使其保留预期的信号方差。

3. 图像质量和生成器平滑度

尽管 GAN 度量标准（例如 FID 或 Precision and Recall（P&R））成功地捕获了生成器的许多方面，但它们仍然在图像质量上处于盲点。例如，请参考图 13 和图 14，它们对比了具有相同 FID 和 P&R 分数但整体质量明显不同的生成器（见底部 2）。

我们观察到了感知的图像质量和感知路径长度（PPL）之间的有趣关系[24]，该指标最初是通过测量在潜在空间中的小扰动下生成的图像之间的平均 LPIPS 距离来量化从潜在空间到输出图像的映射平滑度的指标[49]。再次参考图 13 和 14，较小的 PPL（平滑的生成器映射）似乎与较高的整体图像质量相关，而其他指标则看不到该变化。 图 4 检验了这种相关性是更紧密的

2 我们认为，明显的不一致的关键在于要素空间的特定选择，而不是 FID 或 P&R 的基础。最近发现，使用 ImageNet [35]训练的分类器倾向于将决策更多地基于纹理而不是形状[11]，而人类则强烈关注形状[28]。这在我们的上下文中是有意义的，因为 FID 和 P&R 使用分别来自 InceptionV3 [39]和 VGG-16 [39]的高级特征，这些特征是通过这种方式进行训练的，因此可以预期偏向于纹理检测。这样，具有强烈的猫的纹理的图像可能看起来彼此更相似，比人类观察者所在意的细节还要强，从而部分损害了基于密度的度量（FID）和多方面的覆盖度量（P&R）。

通过在 LSUN CAT 上训练的 StyleGAN 上的 W 上各个点周围的潜在空间采样得出的每幅图像 PPL 分数，可以了解：PPL 分数低实际上表示图像的质量高，反之亦然。图 5a 显示了每个图像 PPL 得分的相应直方图，并揭示了分布的长尾。该模型的总体 PPL 只是每个图像 PPL 得分的预期值。

为何 PPL 值低与图像质量的关联不太明显？我们假设在训练过程中，由于判别器会对残破的图像进行惩罚，因此生成器改进的最直接方法是有效地拉伸产生良好图像的潜在空间区域，这将导致劣质图像被压缩为较小的潜在图像空间快速变化的区域。虽然这可以在短期内提高平均输出质量，但累积的失真会损害训练状态，进而损害最终图像质量。

这种经验相关性表明，在训练过程中通过鼓励低 PPL 来支持平滑的生成器映射可能会改善图像质量，我们在以下情况中会证明这种情况。由于产生的正则化项计算起来有些昂贵，因此我们首先描述适用于所有正则化技术的一般优化。

3.1. 延迟正则化

通常，主要损失函数（例如，逻辑损失[15]）和正则化项（例如，R_1 [30]）被写为单个表达式，因此被同时优化。我们观察到，正规化项的计算频率通常比主要损失函数低得多，从而大大降低了它们的计算成本和整体内存使用量。表 1 中的 C 行显示，每 16 个小批量仅执行一次R_1正则化时，不会造成任何危害，并且我们对新的正则化器也采用了相同的策略。附录 B 给出了实现细节。

3.2. 路径长度正则化

生成器中的路径失真过大显然是不良的局部条件：W 中的任何小区域在被 g 映射时都会被任意挤压和拉伸。与早期工作[33]一致，如果在潜在空间中的每个点处，小的位移都在图像空间中产生相同大小的变化，而与摄动方向无关，则我们认为从潜在空间到图像空间的生成器映射条件良好。

在单个$\mathbf{w} \in \mathcal{W}$处，生成器映射$g(\mathbf{w}) : \mathcal{W} \mapsto \mathcal{Y}$的局部度量比例缩放属性由雅可比矩阵$\mathbf{J}_\mathbf{w} = \partial g(\mathbf{w})/\partial \mathbf{w}$捕获。出于保持向量预期长度（无论方向如何）的动机，我们将正则化定义为：

$$\mathbb{E}_{\mathbf{w},\mathbf{y}\sim\mathcal{N}(0,\mathbf{I})} \left(\left\|\mathbf{J}_\mathbf{w}^T\mathbf{y}\right\|_2 - a\right)^2, \tag{4}$$

图 6.渐进式增长导致“阶段”伪像。 在此示例中，牙齿不遵循姿势，而是与摄像机保持对齐，如蓝线所示。

其中 y 是具有正态分布像素强度的随机图像，而$\mathbf{w} \sim f(\mathbf{z})$，其中 z 是正态分布。我们在附录 C 中显示，在高维上，当$\mathbf{J}_\mathbf{w}$在任何 w 处都是正交的（最大到全局范围）时，该先验会最小化。正交矩阵会保留长度，并且不会沿任何维度压缩。

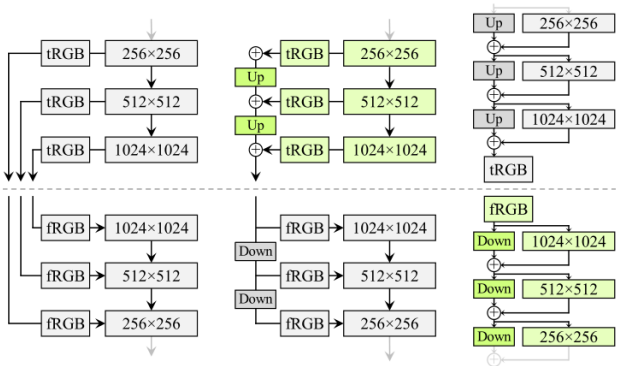
为了避免对雅可比矩阵的显式计算，我们使用恒等式$\mathbf{J}_\mathbf{w}^T\mathbf{y} = \nabla_\mathbf{w}(g(\mathbf{w}) \cdot \mathbf{y})$，它可以使用标准反向传播有效地计算[6]。常数 a 在优化过程中动态设置为长度$\|\mathbf{J}_\mathbf{w}^T\mathbf{y}\|_2$的长期指数移动平均值，从而使优化本身可以找到合适的全局标度。

我们的正则化器与 Odena 等人提出的 Jacobian “夹紧”正则化器密切相关[33]。实际的差异包括我们通过分析计算出乘积$\mathbf{J}_\mathbf{w}^T\mathbf{y}$，而它们使用有限的差异来估计满足$\mathcal{Z} \ni \boldsymbol{\delta} \sim \mathcal{N}(0, \mathbf{I})$的$\mathbf{J}_\mathbf{w}\boldsymbol{\delta}$。应当指出，生成器[45]的频谱归一化[31]仅约束最大奇异值，对其他奇异值没有约束，因此不一定导致更好的调节。

在实践中，我们注意到路径长度正则化导致更可靠和始终如一的行为模型，从而使架构探索更加容易。图 5b 显示，路径长度正则化明显改善了每个图像 PPL 分数的分布。表 1 的 D 行表明，正则化可以按预期方式降低 PPL，但在 LSUN CAR 和结构比 FFHQ 少的其他数据集中，FID 和 PPL 之间需要权衡。此外，我们观察到，更平滑的生成器更易于反转（第 5 节）。

4. 渐进式增长修正

渐进生长[23]在稳定高分辨率图像合成方面已经非常成功，但它会导致其自身的特征失真。关键问题在于，逐渐增长的生成器似乎对细节的位置偏好很高。 随附的视频显示



(a) MSG-GAN (b) Input/output skips (c) Residual nets

图 7.三种生成器（虚线上方）和判别器体系结构。Up 和 Down 分别表示双线性上和下采样。 在残差网络中，这些还包括 1×1 卷积以调整特征图的数量。tRGB 和 fRGB 在 RGB 和高维每像素数据之间转换。 配置 E 和 F 中使用的体系结构以绿色突出显示。

当牙齿或眼睛等特征在图像上平滑移动时，它们可能会停留在原位，然后跳到下一个首选位置。图 6 显示了相关的伪像。我们认为问题在于，在逐步增长的过程中，每个分辨率都会瞬间用作输出分辨率，迫使其生成最大频率细节，然后导致受过训练的网络在中间层具有过高的频率，从而损害了位移不变性[48]。附录 A 显示了一个示例。这些问题促使我们寻找一种替代方法，该方法将保留渐进式增长的好处而消除弊端。

4.1. 可选的网络架构

虽然 StyleGAN 在生成器（合成网络）和判别器中使用简单的前馈设计，但仍有大量工作致力于研究更好的网络体系结构。特别地，在生成方法的背景下，跳跃连接[34、22]，残差网络[17、16、31]和分层方法[7、46、47]也被证明是非常成功的。因此，我们决定重新评估 StyleGAN 的网络设计，并寻找一种能够生成高质量图像而不会逐渐增长的体系结构。

图 7a 显示了 MSG-GAN [22]，它使用多个跳越连接来连接生成器和判别器的匹配分辨率。修改了 MSG-GAN 生成器以输出多维图像（mipmap） [41]而不是图像，并且还为每个真实图像计算了类似的表示形式。在图 7b 中，我们通过上采样并求和对应于不同分辨率的 RGB 输出的贡献来简化此设计。在判别器中

FFHQ	D original		D input skips		D residual	
	FID	PPL	FID	PPL	FID	PPL
G original	4.32	237	4.18	207	3.58	238
G output skips	4.33	149	3.77	**116**	**3.31**	117
G residual	4.35	187	3.96	201	3.79	203

LSUN Car	D original		D input skips		D residual	
	FID	PPL	FID	PPL	FID	PPL
G original	3.75	905	3.23	758	3.25	802
G output skips	3.77	544	3.86	**316**	3.19	471
G residual	3.93	981	3.40	667	**2.66**	645

表 2.没有逐步增长的生成器和判别器架构的比较。生成器与输出跳越和残差判别器的组合对应于主结果表中的配置 E。

我们类似地将下采样的图像提供给判别器的每个分辨率块。我们在所有上采样和下采样操作中都使用了双线性滤波。在图 7c 中，我们进一步修改了设计以使用残差连接（见底部 3）。此设计类似于 LAPGAN [7]，但没有 Denton 等人使用的逐分辨率判别器。

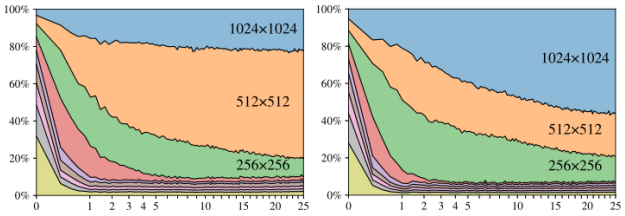
表 2 比较了三种生成器架构和三种判别器架构：StyleGAN 中使用的原始前馈网络，跳越连接和残差网络，所有这些网络都经过训练而没有逐步增长。为 9 种组合中的每一种都提供了 FID 和 PPL。我们可以看到两个广泛的趋势：在所有配置中，跳越生成器中的连接会大大改善 PPL，而残差的判别器网络显然对 FID 有利。后者也许不足为奇，因为判别器可辨别分类器的结构（其中已知残差结构会有所帮助）。但是，残差的体系结构对生成器有害。当两个网络都是残差的时，唯一的例外是 LSUN CAR 中的 FID。

在本文的其余部分，我们使用跳越生成器和残差判别器，而不使用渐进式增长。这对应于表 1 中的配置 E，从表中可以看出，切换到该设置可以显著改善 FID 和 PPL。

4.2. 分辨率用法

我们要保留的渐进式增长的关键方面是，生成器将首先关注低分辨率功能，然后慢慢将注意力转移到更精细的细节上。图 7 中的体系结构使生成器可以以显著方式首先输出不受高分辨率层影响的低分辨率图像，然后随着训练的进行将焦点转移到高分辨率层。由于不会以任何方式强制执行此操作，因此生成器仅在有益时才会执行此操作。

[3] 在残差网络架构中，两条路径的相加会导致信号方差加倍，我们可以通过乘以$1/\sqrt{2}$来抵消。 这对于我们的网络至关重要，而在分类 Resnet [17]中，问题通常被批归一化处理隐藏。



(a) StyleGAN-sized (config E) (b) Large networks (config F)

图 8.每个分辨率对生成器输出的贡献与训练时间的关系。纵轴显示了不同分辨率的相对标准偏差的细目分类，而横轴对应于训练进度，以判别器显示的数百万个训练图像进行测量。我们可以看到，网络一开始就专注于低分辨率图像，并且随着训练的进行逐渐将其注意力转移到更大的分辨率上。在（a）中，生成器基本上输出512^2图像，并对其进行1024^2的较小锐化，而在（b）中，较大的网络更多地关注高分辨率细节。

Dataset	Resolution	StyleGAN (A)		Ours (F)	
		FID	PPL	FID	PPL
LSUN CAR	512×384	3.27	1485	**2.32**	**416**
LSUN CAT	256×256	8.53	924	**6.93**	**439**
LSUN CHURCH	256×256	4.21	742	**3.86**	**342**
LSUN HORSE	256×256	3.83	1405	**3.43**	**338**

表 3.使用 FID 和 PPL 测量的 LSUN 数据集的改进。我们对 CAR 进行了 5700 万幅图像的训练，对 CAT 进行了 88M 的训练，对 CHURCH 进行了 48M 的训练，对 HORSE 进行了 100M 的训练。

为了分析实践中的行为，我们需要量化生成器在训练过程中对特定分辨率的依赖程度。

由于跳越生成器（图 7b）通过显式求和来自多个分辨率的 RGB 值来形成图像，因此我们可以通过测量对应层对最终图像的贡献来估计对应层的相对重要性。在图 8a 中，我们绘制了每个 tRGB 层产生的像素值的标准偏差与训练时间的关系。我们计算 w 的 1024 个随机样本的标准差，并对值进行归一化，以使它们的总和为 100%。

在训练开始时，我们可以看到新的跳越生成器的行为类似于渐进式增长-现在无需更改网络拓扑即可实现。因此，可以预期，最高分辨率将在训练结束时支配。但是，该图表明在实践中这没有发生，这表明生成器可能无法“完全利用”目标分辨率。为了验证这一点，我们手动检查了生成的图像，并注意到它们通常缺少训练数据中存在的某些像素级别的细节-图像可以描述为512^2图像的锐化版本，而不是真实的1024^2图像。

这导致我们假设存在容量问题

在我们的网络中，我们通过将两个网络的最高分辨率层中的特征图的数量加倍来进行测试（见底部 4）。这使行为更加符合预期：图 8b 显示了贡献的显著增加，表 1 的 F 行显示 FID 和 Recall 显著提高。

表 3 在几个 LSUN 类别中比较了 StyleGAN 和我们改进的变体，再次显示了 FID 的明显改进和 PPL 的显著进步。尺寸的进一步增加可能会带来更多好处。

5. 将图像投影到潜在空间

反转合成网络 g 是一个有趣的问题，它具有许多应用。在潜在特征空间中处理给定图像需要首先为其找到匹配的潜在向量 w。另外，随着 GAN 图像质量的提高，将潜在的合成图像归因于生成它的网络变得越来越重要。

先前的研究[1，10]建议，如果不为生成器的每一层选择一个单独的 w，则结果会改善，而不是找到一个共同的潜矢量。在早期的编码器实现中使用了相同的方法[32]。虽然以这种方式扩展潜在空间可以找到与给定图像更接近的匹配，但它还可以投射不应该具有潜在表示的任意图像。着重于对生成图像的取证检测，我们专注于在原始的、未扩展的潜在空间中查找潜在编码，因为这些编码对应于生成器可能生成的图像。

我们的投影方法在两个方面与以前的方法不同。首先，我们在优化过程中向潜在编码添加了降低的噪声，以便更全面地探索潜在空间。其次，我们还优化了 StyleGAN 生成器的随机噪声输入，对它们进行规则化处理以确保它们最终不会携带相干信号。该正则化基于强制噪声图的自相关系数在多个尺度上匹配单位高斯噪声的自相关系数。我们的投影方法的详细信息可以在附录 D 中找到。

5.1. 检测生成的图像

可以训练分类器以相当高的置信度检测 GAN 生成的图像[29、44、40、50]。然而，鉴于进展的迅速，这可能不是持久的情况。基于投影的方法的独特之处在于，它们可以以匹配的潜矢量的形式提供证据，说明图像是由特定网络合成的[2]。也没有任何理由说明其有效性

[4] 我们在64^2–1024^2分辨率中将特征图的数量增加了一倍，同时保持网络的其他部分不变。这样，生成器中可训练参数的总数增加了 22%（25M→30M），而判别器中可训练参数的总数增加了 21%（24M→29M）。

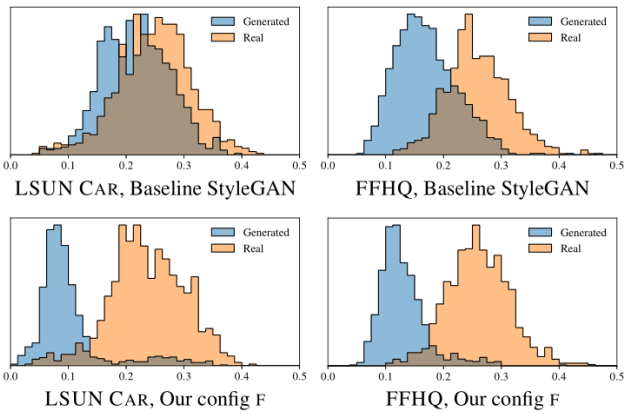


图 9.原始图像和投影图像之间的 LPIPS 距离。生成的图像的距离直方图以蓝色显示，橙色为真实图像。尽管改进后的生成器具有更高的图像质量，但将生成的图像投影到其潜在空间 W 中要容易得多。在所有情况下都使用相同的投影方法。

随着合成图像质量的提高，它会减少，这与基于分类器的方法不同，后者将来可能会遇到较少的特征线索。

事实证明，即使生成的图像质量更高，我们对 StyleGAN 的改进也使得使用基于投影的方法更容易检测生成的图像。我们通过计算原始图像和重新合成图像之间的 LPIPS [49]距离$D_{\mathrm{LPIPS}}[\boldsymbol{x}, g(\tilde{g}^{-1}(\boldsymbol{x}))]$来测量投影成功的程度，其中 x 是要分析的图像，而$\tilde{g}^{-1}$表示近似投影操作。图 9 显示了使用原始 StyleGAN 和我们的最佳架构的 LSUN CAR 和 FFHQ 数据集的这些距离的直方图，图 10 显示了示例投影。如后者所示，使用我们改进的架构生成的图像可以很好地投影到生成器输入中，从而可以明确地将它们归因于生成网络。使用原始 StyleGAN，即使从技术上来说应该可以找到匹配的潜矢量，但实际上，潜空间似乎太复杂了，无法可靠地成功实现。我们具有更好的潜在空间 W 的改进模型，因此受此问题的影响要小得多。

6. 结论与未来工作

我们已经确定并修复了 StyleGAN 中的多个图像质量问题，从而进一步改善了质量，并在多个数据集中大大提高了先进技术水平。在某些情况下，如所附视频所示，在运动中可以更清楚地看到这些改进。附录 A 包含使用我们的方法可获得的结果的更多示例。尽管质量有所提高，但与原始 StyleGAN 相比，使用基于投影的方法检测由我们的方法生成的图像更容易。

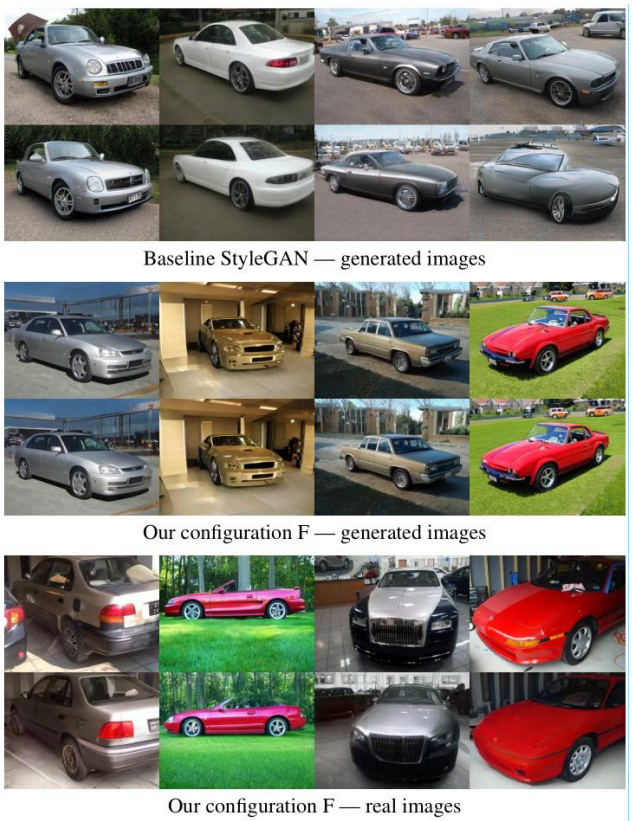


图 10.示例图像及其投影和重新合成的对应图像。对于每种配置，第一行显示目标图像，第二行显示相应的投影潜在矢量和噪声输入的合成。顶部：使用基线 StyleGAN，投影通常可以找到与生成的图像相当接近的匹配，但尤其是背景不同于原始图像。中：使用我们最好的架构生成的图像几乎可以完美地投影回生成器输入中，从而可以明确地归因于生成模型。下图：投影的真实图像（来自训练集）显示出与原始图像明显的差异，正如预期的那样。所有测试均使用相同的投影方法和超参数完成。

训练分数也有所提高。在1024^2分辨率下，原始 StyleGAN（表 1 中的配置 A）在配备 8 个 Tesla V100 GPU 的 NVIDIA DGX-1 上以每秒 37 张图像的速度训练，而我们的配置 E 以 61 图片/秒的速度训练 40%。大多数加速来自权重解调，延迟正则化和代码优化带来的简化数据流。Config F（大型网络）的训练速度为 31 图片/秒，因此，其训练成本仅轻微高于原始 StyleGAN。使用配置 F，FFHQ 的总训练时间为 9 天，而 LSUN CAR 的总训练时间为 13 天。

作为未来的工作，研究进一步改善路径长度正则化可能是富有成果的，例如，通过用数据驱动的特征空间度量替换像素空间 L2 距离。

7. 致谢

我们感谢 Liu Ming-Yu 的早期评论，Timo Viitanen 的代码发布帮助，以及 Tero Kuosmanen 的计算基础架构。

参考文献

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2StyleGAN: How to embed images into the StyleGAN latent space? In ICCV, 2019. 7

[2] Michael Albright and Scott McCloskey. Source generator attribution via inversion. In CVPR Workshops, 2019. 7

[3] Carl Bergstrom and Jevin West. Which face is real? http://www.whichfaceisreal.com/learn.html, Accessed November 15, 2019. 1

[4] Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer, 2006. 17

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. CoRR, abs/1809.11096, 2018. 1

[6] Yann N. Dauphin, Harm de Vries, and Yoshua Bengio. Equi-librated adaptive learning rates for non-convex optimization. CoRR, abs/1502.04390, 2015. 5

[7] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Robert Fergus. Deep generative image models using a Laplacian pyramid of adversarial networks. CoRR, abs/1506.05751, 2015. 6

[8] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. Distill, 2018. https://distill.pub/2018/feature-wise-transformations. 1

[9] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kud-lur. A learned representation for artistic style. CoRR, abs/1610.07629, 2016. 1

[10] Aviv Gabbay and Yedid Hoshen. Style generator inversion for image enhancement and animation. CoRR, abs/1906.11880, 2019. 7

[11] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increas-ing shape bias improves accuracy and robustness. CoRR, abs/1811.12231, 2018. 1, 4

[12] Golnaz Ghiasi, Honglak Lee, Manjunath Kudlur, Vincent Dumoulin, and Jonathon Shlens. Exploring the structure of a real-time, arbitrary neural artistic stylization network. CoRR, abs/1705.06830, 2017. 1

[13] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, pages 249–256, 2010. 3

[14] G.H. Golub and C.F. Van Loan. Matrix Computations. Johns Hopkins Studies in the Mathematical Sciences. Johns Hop-kins University Press, 2013. 16

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In NIPS, 2014. 1, 5, 10

[16] Ishaan Gulrajani, Faruk Ahmed, Mart´ın Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of Wasserstein GANs. CoRR, abs/1704.00028, 2017. 6

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015. 6

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level perfor-mance on ImageNet classification. CoRR, abs/1502.01852, 2015. 3

[19] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local Nash equi-librium. In Proc. NIPS, pages 6626–6637, 2017. 1

[20] Xun Huang and Serge J. Belongie. Arbitrary style trans-fer in real-time with adaptive instance normalization. CoRR, abs/1703.06868, 2017. 1

[21] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal co-variate shift. CoRR, abs/1502.03167, 2015. 2

[22] Animesh Karnewar, Oliver Wang, and Raghu Sesha Iyengar. MSG-GAN: multi-scale gradient GAN for stable image syn-thesis. CoRR, abs/1903.06048, 2019. 6

[23] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. CoRR, abs/1710.10196, 2017. 1, 5, 10

[24] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proc. CVPR, 2018. 1, 2, 4, 10, 12, 16

[25] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In ICLR, 2015. 10, 19

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet classification with deep convolutional neural net-works. In NIPS, pages 1097–1105. 2012. 15

[27] Tuomas Kynka¨anniemi,¨ Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall met-ric for assessing generative models. In Proc. NeurIPS, 2019. 1, 2, 3

[28] Barbara Landau, Linda B. Smith, and Susan S. Jones. The importance of shape in early lexical learning. Cognitive De-velopment, 3(3), 1988. 4

[29] Haodong Li, Han Chen, Bin Li, and Shunquan Tan. Can forensic detectors identify GAN generated images? In Proc. Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2018. 7

[30] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? CoRR, abs/1801.04406, 2018. 5, 10

[31] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. CoRR, abs/1802.05957, 2018. 1, 5, 6

[32] Dmitry Nikitko. StyleGAN – Encoder for official TensorFlow implementation. https://github.com/Puzer/stylegan-encoder/, 2019. 7

[33] Augustus Odena, Jacob Buckman, Catherine Olsson, Tom B. Brown, Christopher Olah, Colin Raffel, and Ian Goodfellow. Is generator conditioning causally related to GAN perfor-mance? CoRR, abs/1802.08768, 2018. 5, 18

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmen-tation. In Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI), pages 234–241, 2015. 6

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, San-jeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. ImageNet large scale visual recognition chal-lenge. In Proc. CVPR, 2015. 4

[36] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. CoRR, abs/1806.00035, 2018. 1

[37] Tim Salimans and Diederik P. Kingma. Weight normaliza-tion: A simple reparameterization to accelerate training of deep neural networks. CoRR, abs/1602.07868, 2016. 3

[38] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Inter-preting the latent space of GANs for semantic face editing. CoRR, abs/1907.10786, 2019. 1

[39] Karen Simonyan and Andrew Zisserman. Very deep convo-lutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014. 1, 4

[40] Run Wang, Lei Ma, Felix Juefei-Xu, Xiaofei Xie, Jian Wang, and Yang Liu. FakeSpotter: A simple baseline for spotting AI-synthesized fake faces. CoRR, abs/1909.06122, 2019. 7

[41] Lance Williams. Pyramidal parametrics. SIGGRAPH Com-put. Graph., 17(3):1–11, 1983. 6

[42] Sitao Xiang and Hao Li. On the effects of batch and weight normalization in generative adversarial networks. CoRR, abs/1704.03971, 2017. 3

[43] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianx-iong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. CoRR, abs/1506.03365, 2015. 10

[44] Ning Yu, Larry Davis, and Mario Fritz. Attributing fake im-ages to GANs: Analyzing fingerprints in generated images. CoRR, abs/1811.08180, 2018. 7

[45] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augus-tus Odena. Self-attention generative adversarial networks. CoRR, abs/1805.08318, 2018. 5

[46] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris N. Metaxas. Stack-GAN: text to photo-realistic image synthesis with stacked generative adversarial networks. In ICCV, 2017. 6

[47] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiao-gang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stack-GAN++: realistic image synthesis with stacked generative adversarial networks. CoRR, abs/1710.10916, 2017. 6

[48] Richard Zhang. Making convolutional networks shift-invariant again. In Proc. ICML, 2019. 6, 10

[49] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shecht-man, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In Proc. CVPR, 2018. 4, 8, 19

[50] Xu Zhang, Svebor Karaman, and Shih-Fu Chang. Detect-ing and simulating artifacts in GAN fake images. CoRR, abs/1907.06515, 2019. 7

A. 图片质量

我们包括几张大图像，这些图像说明了与图像质量有关的各个方面。图 11 显示了一些示例，这些示例说明了使用我们的方法在 FFHQ 中可以实现的质量和多样性，而图 12 显示了本文提到的所有数据集的未经评估的结果。

图 13 和图 14 展示了 FID 和 P&R 给出非直观结果的情况，但 PPL 似乎更符合人类的判断。

我们还包括与 StyleGAN 文物有关的图像。图 15 显示了一种罕见情况，其中在 StyleGAN 激活中无法出现斑点伪像，从而导致图像严重损坏。图 16 将表 1 的配置 A 和 F 中的激活可视化。很明显，渐进式增长导致中间层的频率较高，从而损害了网络的移位不变性。我们假设使用渐进式增长会导致观察到的位置偏向于细节。

B. 实现细节

我们在对应于表 1 中配置 A 的 StyleGAN 官方 TensorFlow 实现（见底部 5）的基础上实现了我们的技术。我们保持大多数细节不变，包括 Z 和 W 的维数（512），映射网络体系结构（8 个完全连接层的布局），100×更低的学习率，所有可训练参数的学习率均等[23]，Leaky ReLU 激活选用参数 α=0.2，所有上/下采样层的双线性滤波[48] [24]，小批量标准偏差层在判别器[23]的末尾，生成器权重的指数移动平均值[23]，样式混合正则化[24]，具有 R1 正则化的非饱和逻辑损失[15] [30]，具有相同的超参数（$\beta_1 = 0$; $\beta_2 = 0.99$; $\epsilon = 10^{-8}$; minibatch = 32）和训练数据集[24，43]。我们使用 TensorFlow 1.14.0 和 cuDNN 7.4.2 在配备 8 个 Tesla V100 GPU 的 NVIDIA DGX-1 上进行了所有训练。

生成器重新设计 在配置 B–F 中，我们用经过修订的体系结构替换了原来的 StyleGAN 生成器。除了第 2 节中突出显示的更改之外，我们还使用$\mathcal{N}(0, 1)$初始化常量输入c_1的组件，并简化了噪声广播操作，以对所有特征图使用单一共享比例因子。除了输出层（图 7 中的 tRGB）不进行解调外，我们在所有卷积层中都使用了权重调制和解调。在具有1024^2输出分辨率的情况下，生成器总共包含 18 个仿射变换层，其中第一个对应于4^2分辨率，接下来的两个对应于8^2分辨率，依此类推。

[5] https://github.com/NVlabs/stylegan

图 11.四个精选示例说明了使用我们的方法（配置 F）可获得的图像质量和多样性。

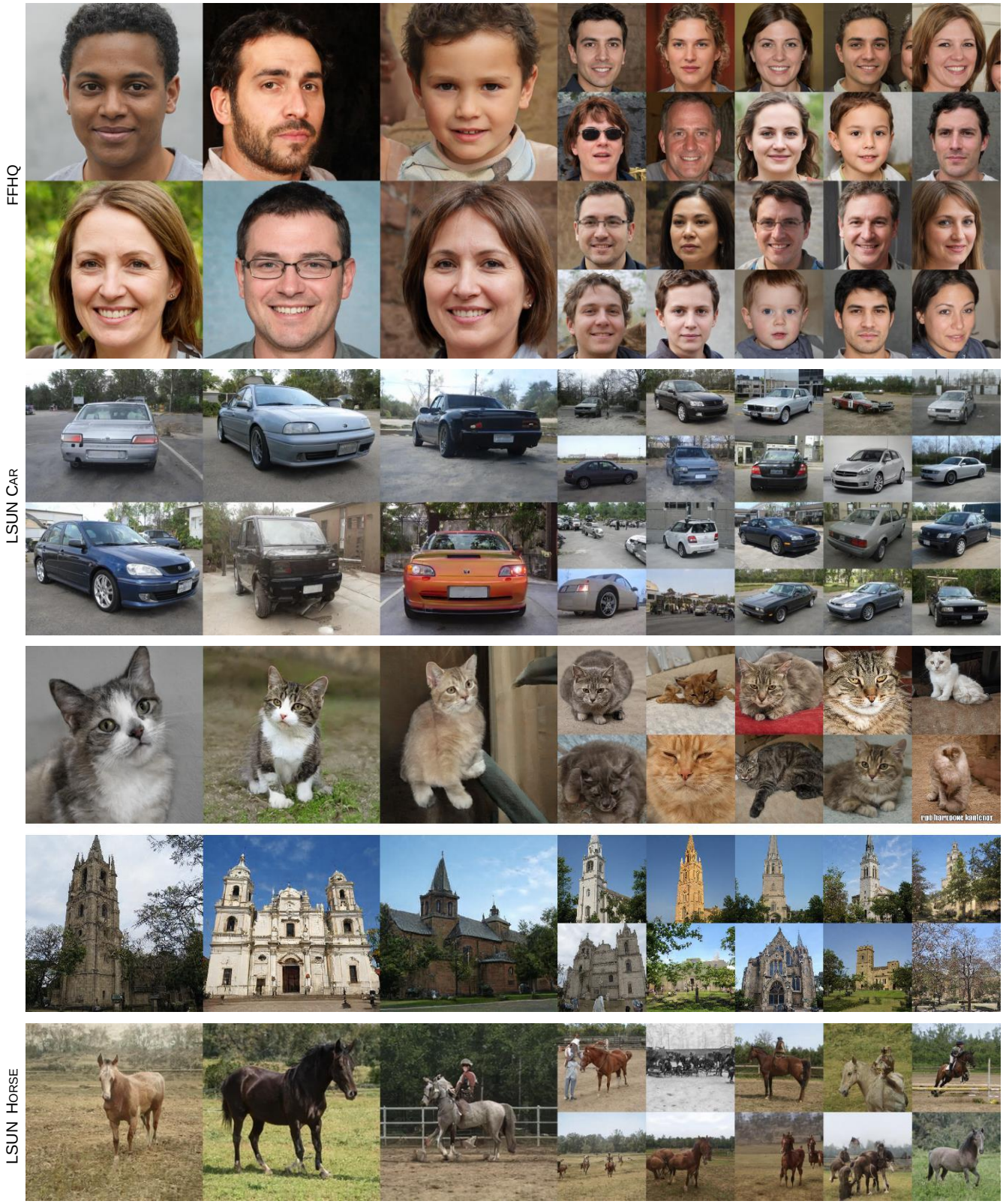
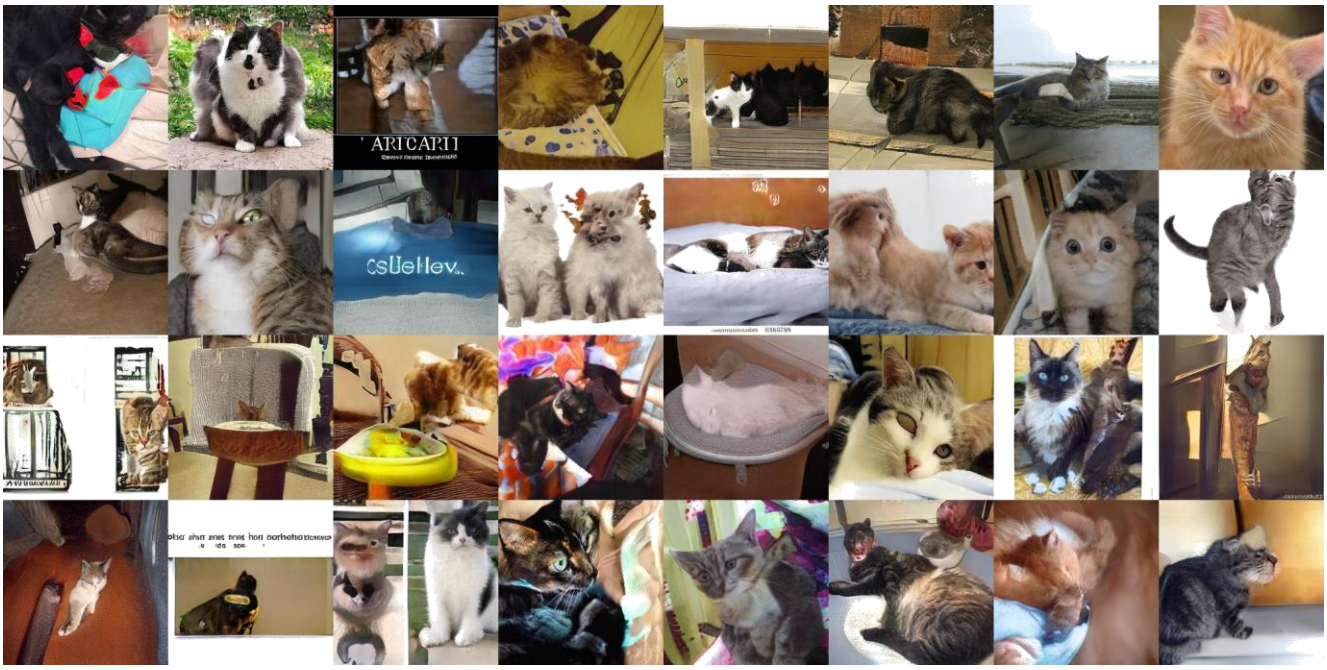
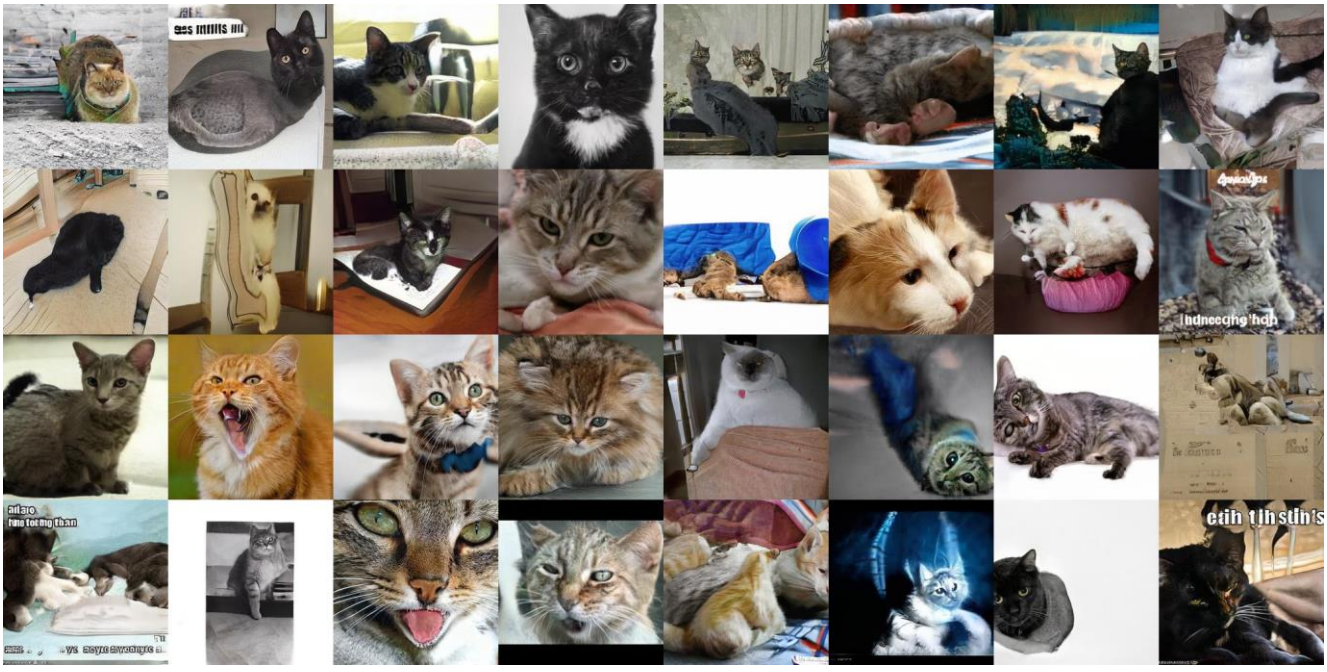


图 12.表 1 和表 3 中使用的每个数据集的未整理结果。图像对应于我们的生成器（配置 F）产生的随机输出，在所有分辨率下均使用ψ = 0.5 进行了截断[24]。

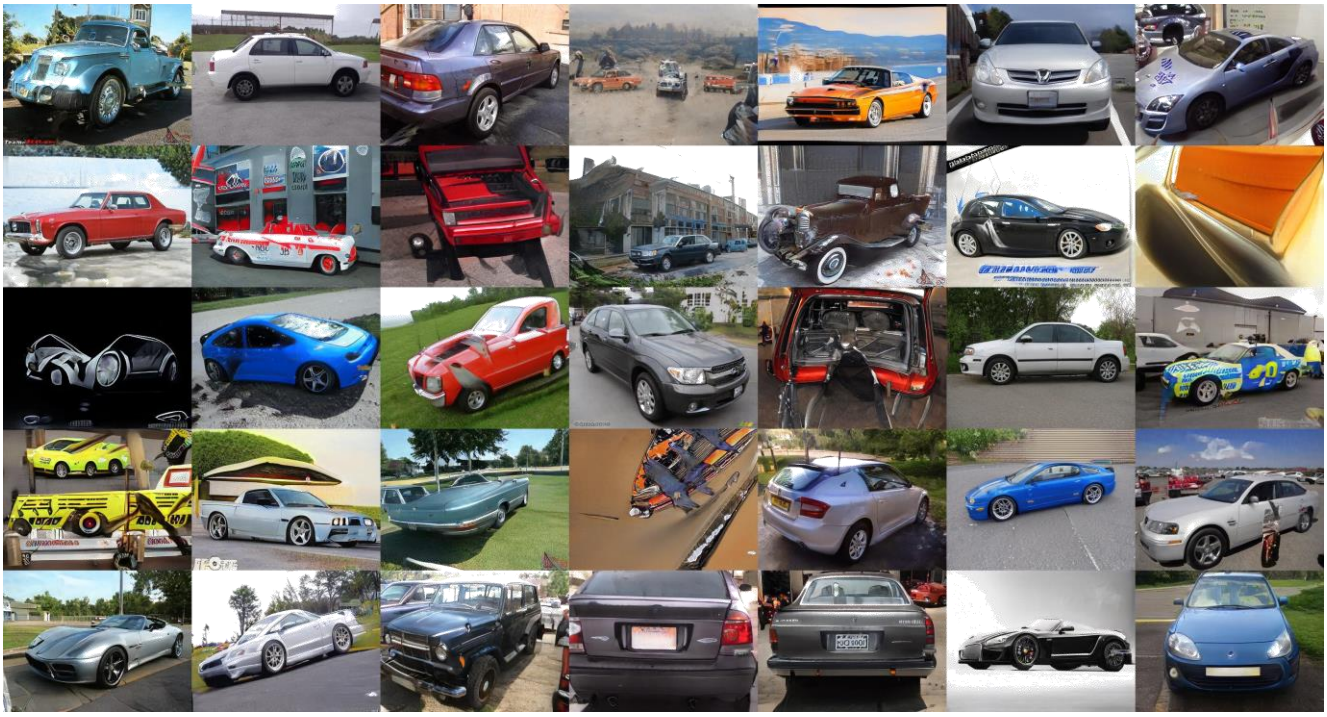


Model 1: FID = 8.53, P = 0.64, R = 0.28, PPL = 924

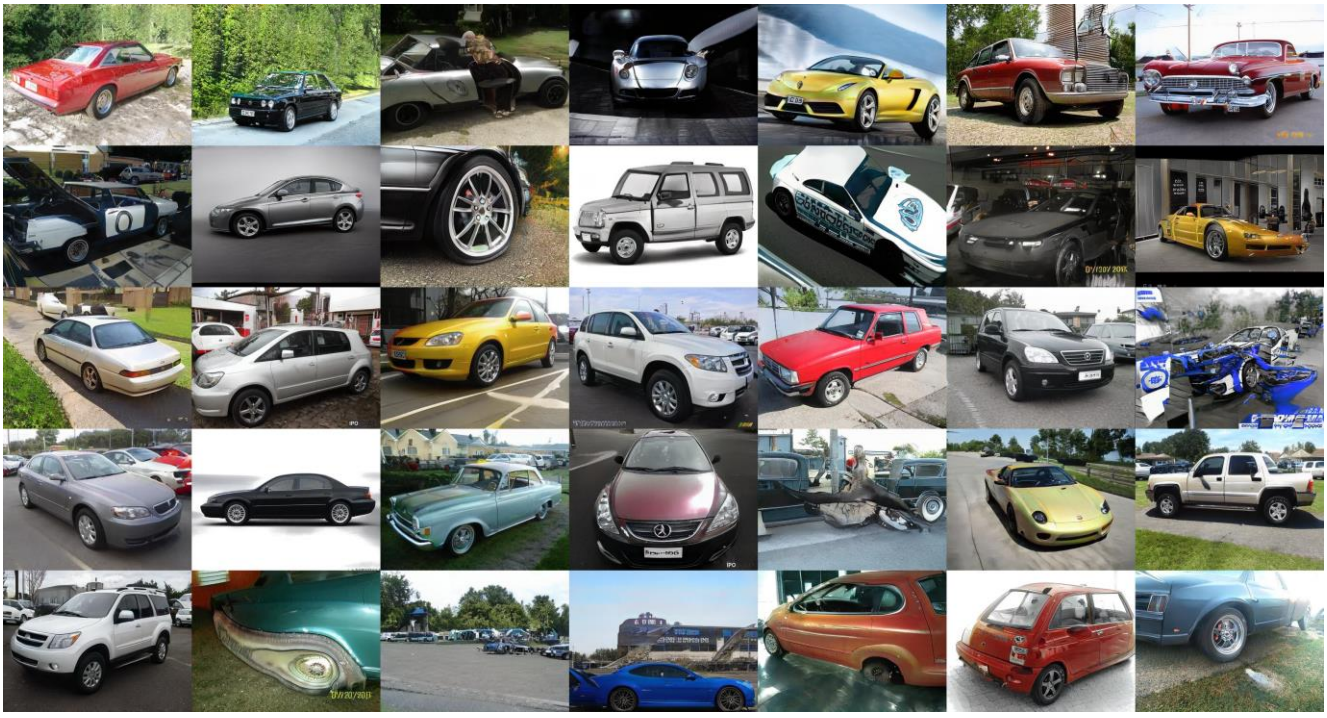


Model 2: FID = 8.53, P = 0.62, R = 0.29, PPL = 387

图 13.来自未经截断的 LSUN CAT 训练的两个生成模型的未经整理的示例。 模型 1 和 2 的 FID，精度和召回率相似，尽管模型 1 和 2 会更频繁地产生猫形物体。 感知路径长度（PPL）表示更偏爱模型 2。模型 1 对应于表 3 中的配置 A，模型 2 是配置 F 的早期训练快照。



Model 1: FID = 3.27, P = 0.70, R = 0.44, PPL = 1485



Model 2: FID = 3.27, P = 0.67, R = 0.48, PPL = 437

图 14.在未经截断的 LSUN CAR 上训练的两个生成模型的未经整理的示例。 模型 1 和 2 的 FID，精度和召回率相似，尽管模型 1 和 2 会更频繁地产生汽车形状的物体。 感知路径长度（PPL）表示更偏爱模型 2。模型 1 对应于表 3 中的配置 A，模型 2 是配置 F 的早期训练快照。

图 15。StyleGAN 生成器中液滴伪影的重要性示例。 我们比较了两个生成的图像，一个成功，另一个严重损坏。 使用实例归一化将相应的特征图归一化为可见的动态范围。对于顶部图像，液滴伪影以64^2分辨率开始形成，在128^2中清晰可见，并逐渐以更高的分辨率主导特征图。对于底部图像，64^2在质量上与顶部行相似，但在128^2中不会出现液滴。因此，在归一化特征图中，面部特征更强。这导致256^2中的过冲（overshoot），随后在后续分辨率中形成多个杂散液滴。根据我们的经验，StyleGAN 图像中很少会丢失液滴，并且实际上生成器完全依赖于液滴的存在。

(a) Progressive growing (config A)　　(b) Without progressive growing (config F)

图 16.渐进式增长导致中间层中的频率成分明显更高。这会损害网络的平移不变性，并使得在高分辨率层中精确定位要素变得更加困难。

权重解调 考虑到等式 1 和 3 的实际实现，必须注意，对于小批量中的每个样例，所得的权重集将有所不同，这排除了使用标准卷积结构的直接实现。相反，我们选择采用分组卷积[26]，该卷积最初是作为一种通过将输入特征图划分为多个独立的组（每个组都有各自专用的权重）来减少计算成本的方法。我们通过临时调整权重和激活来实现等式 1 和 3，以便每个卷积都能看到一个样本有 N 组-而不是 N 个样本一组。这种方法非常高效，因为整形（reshape）操作实际上并未修改权重和激活张量的内容。

延迟正则化 在配置 C-F 中，我们通过在单独的正则化中评估正则化项（R1 和路径长度）来采用延迟正则化（3.1 节），我们每 k 次训练迭代执行一次。我们在主要损失和正则化项之间共享 Adam 优化器的内部状态，以便优化器

首先从 k 次迭代的主要损失中看到梯度，然后从一次迭代的正则项中看到梯度。为了补偿我们现在执行 k + 1 训练迭代而不是 k 的事实，我们调整了优化器超参数$\lambda' = c \cdot \lambda$，$\beta_1' = (\beta_1)^c$和$\beta_2' = (\beta_2)^c$，其中$c = k/(k+1)$。我们还将正则项乘以 k 来平衡其梯度的整体大小。我们将 k = 16 用作判别器，将 k = 8 用作生成器。

路径长度正则化 配置 D–F 包括我们新的路径长度正则化器（第 3.2 节）。我们将目标标度 a 初始化为零，并使用衰减系数$\beta_{\text{pl}} = 0.99$在每个 GPU 上将其作为$\|\mathbf{J}_{\mathbf{w}}^T\mathbf{y}\|_2$的指数移动平均值进行跟踪。我们将正则化项加权为：

$$\gamma_{\text{pl}} = \frac{\ln 2}{r^2(\ln r - \ln 2)}, \tag{5}$$

其中 r 指定输出分辨率（例如 r = 1024）。我们发现这些参数选择可以在所有配置和数据集中可靠地工作。为了确保我们的正则化器与样式混合正则化正确交互，我们将其计算为合成网络所有各个层的平均值。附录 C 详细分析了我们的正则化器对 W 和图像空间之间映射的影响。

渐进生长 在 A–D 配置中，我们使用与 Karras 等人相同的参数进行渐进式生长[24]（从8^2分辨率开始，学习率 λ=10^{-3}，为每个分辨率训练 600k 图像，然后在下一次重新分解时淡出 600k 图像，将学习速率逐渐提高 3×）。在 E–F 配置中，我们禁用渐进式增长，并将学习率设置为固定值$\lambda = 2 \cdot 10^{-3}$，我们发现这可以提供最佳结果。此外，如第 4.1 节中所述，我们在生成器中使用跳越输出，在判别器中使用残差连接。

特定于数据集的调整 与 Karras 等类似[24]，我们通过水平翻转来增强 FFHQ 数据集，以有效地将训练图像的数量从 70k 增加到 140k，并且我们不对 LSUN 数据集进行任何增强。我们发现，训练长度和 R1 正则化权重的最佳选择在不同的数据集和配置之间往往会有很大的差异。除表 1 中的配置 E 以及表 3 中的 LSUN CHURCH 和 LSUN HORSE 外，我们对所有训练使用γ = 10，在表 3 中使用γ =100。进一步的调整可能会带来更多好处。

性能优化 我们对训练进行了广泛的分析，发现在我们的案例中，用于图像过滤，上/下采样，添加偏置和 Leaky ReLU 的默认结构在训练时间和 GPU 内存占用方面具有惊人的高开销。这促使我们使用手写 CUDA 内核来优化这些操作。我们将滤波的上/下采样实现为单个融合操作，将偏置和激活实现为另一个。在1024^2分辨率的配置 E 中，我们的优化使整体训练时间缩短了约 30%，并将内存占用减少了约 20%。

C. 路径长度正则化的作用

3.2 节中描述的路径长度调节器的形式为：

$$\mathcal{L}_{\text{pl}} = \mathbb{E}_{\mathbf{w}}\mathbb{E}_{\mathbf{y}}\left(\left\|\mathbf{J}_{\mathbf{w}}^T\mathbf{y}\right\|_2 - a\right)^2, \tag{6}$$

其中$\mathbf{y} \in \mathbb{R}^M$是生成的图像空间中的单位正态分布随机变量（尺寸$M = 3wh$，即 RGB 图像尺寸），$\mathbf{J}_{\mathbf{w}} \in \mathbb{R}^{M \times L}$是生成函数$g : \mathbb{R}^L \mapsto \mathbb{R}^M$在潜在的空间点$\mathbf{w} \in \mathbb{R}^L$的雅可比矩阵，并且$a \in \mathbb{R}$是表示梯度所需比例的全局变量。

C.1. 逐像素雅可比矩阵的影响

当分别在每个潜在空间点 w 上使 y 的内部期望最小时，该先验值将最小化。 在本小节中，我们显示当雅可比矩阵$\mathbf{J}_{\mathbf{w}}$正交时，内部期望被（近似）最小化，直到全局缩放因子。一般策略是利用众所周知的事实，即在高维度 L 上，单位正态分布的密度集中在半径为$\sqrt{L}$的球壳上。然后，当矩阵$\mathbf{J}_{\mathbf{w}}^T$将期望函数按比例缩放为在这个半径上有其最小值。这可以通过任何正交矩阵（每个 w 都具有合适的全局标度）来实现。

我们首先考虑内在期望

$$\mathcal{L}_{\mathbf{w}} := \mathbb{E}_{\mathbf{y}}\left(\left\|\mathbf{J}_{\mathbf{w}}^T\mathbf{y}\right\|_2 - a\right)^2.$$

首先，我们注意到 y 分布的径向对称性以及l_2范数的径向对称性使我们仅关注于对角矩阵。使用奇异值分解$\mathbf{J}_{\mathbf{w}}^T = \mathbf{U}\tilde{\mathbf{\Sigma}}\mathbf{V}^T$可以看出这一点，其中$\mathbf{U} \in \mathbb{R}^{L \times L}$和$\mathbf{V} \in \mathbb{R}^{M \times M}$是正交矩阵，而$\tilde{\mathbf{\Sigma}} = [\mathbf{\Sigma}\ \mathbf{0}]$是对角矩阵$\mathbf{\Sigma} \in \mathbb{R}^{L \times L}$和零矩阵$\mathbf{0} \in \mathbb{R}^{L \times (M-L)}$的水平串联[14]。由于通过正交矩阵旋转单位正态随机变量会使分布保持不变，而旋转向量将使其范数保持不变，

因此该表达式简化为

$$\begin{aligned}\mathcal{L}_{\mathbf{w}} &= \mathbb{E}_{\mathbf{y}}\left(\left\|\mathbf{U}\tilde{\mathbf{\Sigma}}\mathbf{V}^T\mathbf{y}\right\|_2 - a\right)^2 \\ &= \mathbb{E}_{\mathbf{y}}\left(\left\|\tilde{\mathbf{\Sigma}}\mathbf{y}\right\|_2 - a\right)^2.\end{aligned}$$

此外，$\tilde{\mathbf{\Sigma}}$ 中的零矩阵将 y 的维度降到 L 之外，从而有效地边缘化了它们在这些维度上的分布。边缘化分布还是剩余 L 维上的单位正态分布。然后让我们考虑表达式的最小化

$$\mathcal{L}_{\mathbf{w}} = \mathbb{E}_{\tilde{\mathbf{y}}}\left(\|\mathbf{\Sigma}\tilde{\mathbf{y}}\|_2 - a\right)^2,$$

在对角平方矩阵 $\mathbf{\Sigma} \in \mathbb{R}^{L\times L}$ 上，其中 $\tilde{\mathbf{y}}$ 是维度 L 的正态分布。总而言之，所有共享 $\mathbf{\Sigma}$ 的相同奇异值的矩阵 $\mathbf{J}_{\mathbf{w}}^T$ 对于原始损失函数都会产生相同的值。

接下来，我们表明，当对角矩阵 $\mathbf{\Sigma}$ 在每个对角线入口处都具有特定的相同值时，即表达式为单位矩阵的常数倍时，该表达式将被最小化。我们首先将期望写为 $\tilde{\mathbf{y}}$ 的概率密度的整数：

$$\begin{aligned}\mathcal{L}_{\mathbf{w}} &= \int \left(\|\mathbf{\Sigma}\tilde{\mathbf{y}}\|_2 - a\right)^2 p_{\tilde{\mathbf{y}}}(\tilde{\mathbf{y}})\,\mathrm{d}\tilde{\mathbf{y}} \\ &= (2\pi)^{-\frac{L}{2}} \int \left(\|\mathbf{\Sigma}\tilde{\mathbf{y}}\|_2 - a\right)^2 \exp\left(-\frac{\tilde{\mathbf{y}}^T\tilde{\mathbf{y}}}{2}\right)\,\mathrm{d}\tilde{\mathbf{y}}\end{aligned}$$

观察密度的径向对称形式，我们变为极坐标 $\tilde{\mathbf{y}} = r\phi$，其中 $r \in \mathbb{R}_+$ 是到原点的距离，$\phi \in \mathbb{S}^{L-1}$ 是单位矢量，即 L -1 维单位球面上的点。变量的这种变化引入了雅可比因子 r^{L-1}：

$$\tilde{\mathcal{L}}_{\mathbf{w}} = (2\pi)^{-\frac{L}{2}} \int_{\mathbb{S}}\int_0^{\infty} \left(r\,\|\mathbf{\Sigma}\phi\|_2 - a\right)^2 r^{L-1} \exp\left(-\frac{r^2}{2}\right)\,\mathrm{d}r\,\mathrm{d}\phi$$

则概率密度 $(2\pi)^{-L/2} r^{L-1} \exp\left(-\frac{r^2}{2}\right)$ 然后是一个以极坐标表示的 L 维单位法向密度，仅取决于半径而不取决于角度。泰勒近似的标准论证表明，当 L 为高时，对于任何密度φ，密度都可以很好地近似为 $(2\pi e/L)^{-L/2}\exp\left(-\frac{1}{2}(r-\mu)^2/\sigma^2\right)$，这是一个（未归一化的）一维 r 的标准密度，以标准差 $\sigma = 1/\sqrt{2}$ 的中心为 $\mu = \sqrt{L}$ [4]。换句话说，L 维单位正态分布的密度集中在半径为 $\sqrt{L}$ 的壳上。将该密度代入积分，损失变为

大约

$$\mathcal{L}_{\mathbf{w}} \approx (2\pi e/L)^{-L/2} \int_{\mathbb{S}}\int_0^{\infty} \left(r\,\|\mathbf{\Sigma}\phi\|_2 - a\right)^2 \exp\left(-\frac{\left(r-\sqrt{L}\right)^2}{2\sigma^2}\right)\,\mathrm{d}r\,\mathrm{d}\phi, \quad (7)$$

在无穷大尺寸 L 的范围内近似变得精确。

为了最大程度地减少这种损失，我们设置 $\mathbf{\Sigma}$ 的函数 $\left(r\,\|\mathbf{\Sigma}\phi\|_2 - a\right)^2$ 表示在半径为 $\sqrt{L}$ 的球壳上获得最小值。这可以通过 $\mathbf{\Sigma} = \frac{a}{\sqrt{L}}\mathbf{I}$ 来实现，其中函数变为 ϕ 的常数，表达式简化为

$$\mathcal{L}_{\mathbf{w}} \approx (2\pi e/L)^{-L/2}\mathcal{A}(\mathbb{S})a^2L^{-1} \int_0^{\infty} \left(r-\sqrt{L}\right)^2 \exp\left(-\frac{\left(r-\sqrt{L}\right)^2}{2\sigma^2}\right)\,\mathrm{d}r,$$

其中 $\mathcal{A}(\mathbb{S})$ 是单位球的表面积（和其他常数一样，与最小化无关）。请注意，抛物线 $(r-\sqrt{L})^2$ 的零点与概率密度的最大值重合，因此，这种 $\mathbf{\Sigma}$ 的函数的选择对等式 7 的内积分的每个 ϕ 最小化。

总而言之，我们证明了——假设潜空间的高维 L-当生成器的雅可比矩阵的所有奇异值都等于一个全局整数时，路径长度的先验（方程 6）的值会最小化常数，在每个潜在空间点 w 上它们是正交的，直到全局常数范围。

虽然理论上α只是在不更改特征属性的情况下缩放特征的值，并且可以将其设置为固定值（例如 1），但实际上它确实会影响训练的动力。如果强加的比例尺与网络的随机初始化导致的比例尺不匹配，则训练将花费其关键的早期步骤将权重推向所需的总体大小，而不是强制执行实际的目标。这可能会降低网络权重的内部状态，并在以后的训练中导致次优性能。根据经验，我们发现设定固定比例会降低训练结果和数据集之间训练结果的一致性。取而代之的是，我们根据雅可比现有规模的运行平均值 $a \approx \mathbb{E}_{\mathbf{w},\mathbf{y}}\left(\left\|\mathbf{J}_w^T\mathbf{y}\right\|_2\right)$ 动态设置α。通过这种选择，先验目标将局部雅可比矩阵的规模朝向已经存在的全局平均值，而不是强迫特定的全局平均值。这也消除了测量雅各比矩阵的适当规模的需要

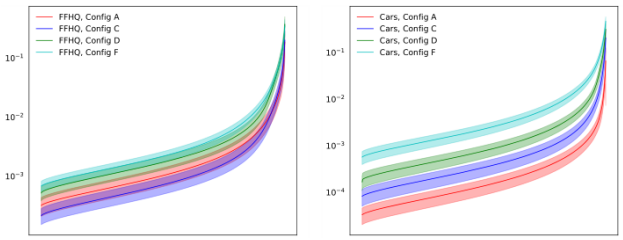


图 17.在随机潜在空间点 w 上评估的雅可比矩阵排序奇异值大小的均值和标准差，最大特征值归一化为 1。在两个数据集中，路径长度正则化（Config D）和新颖架构（Config F）表现出更好的调理效果； 值得注意的是，这种影响在包含更多可变性的 Cars 数据集中更为明显，并且路径长度规则化对 PPL 度量的影响相对较强（表 1）。

就像 Odena 等人所做的那样[33]，考虑了相关的条件。

图 17 显示了在经过和未经过路径长度正则化训练的网络中，根据经验测得的雅可比矩阵奇异值的大小。虽然没有达到正交性，但正则化网络的特征值彼此更接近，这意味着更好的调节，并且效果的强度与 PPL 度量相关（表 1）。

C.2. 对生成器映射的全局属性的影响

在前面的小节中，我们发现先验鼓励生成器映射的雅可比行列式在所有地方都是正交的，尽管图 17 显示了映射实际上并不能完全满足此约束，但是考虑约束对于映射所暗示的全局属性是有启发性的。不失一般性，我们假设矩阵的单位全局标度可以简化演示。

关键特性是，到处都是正交雅可比矩阵的映射 $g : \mathbb{R}^L \mapsto \mathbb{R}^M$ 保留了曲线的长度。要看到这一点，让 $u : [t_0, t_1] \mapsto \mathbb{R}^L$ 参数化潜在空间中的曲线。通过生成器 g 映射曲线，我们在图像空间中获得了一条曲线 $\tilde{u} = g \circ u$。它的弧长是

$$L = \int_{t_0}^{t_1} |\tilde{u}'(t)| \, \mathrm{d}t, \tag{8}$$

其中素数表示相对于 t 的导数。按照链式法则，这等于

$$L = \int_{t_0}^{t_1} |J_g(u(t))u'(t)| \, \mathrm{d}t, \tag{9}$$

其中 $J_g \in \mathbb{R}^{L \times M}$ 是在 u(t)求得的 g 的雅可比矩阵。根据我们的假设，雅可比矩阵是正交的，并且导致向量 $u'(t)$ 的 2-范数不受影响：

$$L = \int_{t_0}^{t_1} |u'(t)| \, \mathrm{d}t. \tag{10}$$

这是在用 g 映射之前在潜在空间中曲线 u 的长度。因此，u 和 $\tilde{u}$ 的长度相等，所以 g 保留了任何曲线的长度。

用微分几何学的语言，g 等距地将欧几里得潜在空间 $\mathbb{R}^L$ 嵌入到 $\mathbb{R}^M$ 的子流形 M 中，例如，代表面孔的图像流形被嵌入所有可能的 RGB 图像的空间中。等距的结果是，潜在空间中的直线段映射到图像流形上的测地线或最短路径：连接两个潜伏空间点的直线 v 不能更短，因此也不能更短对应图像之间的流形图像空间路径 $g \circ v$。例如，人脸图像流形上的测地线是两个人脸之间的连续变体，产生最小的总变化量（按 RGB 空间的 l_2 距离）在每一步中导致图像差异的变形。

如前面小节中的经验实验所示，在实践中未实现等轴测图。训练的全部损失函数是潜在冲突标准的组合，尚不清楚真正的等距映射是否能够表达感兴趣的图像流形。然而，使映射尽可能地等距的压力具有符合期望的结果。特别是，它会阻止不必要的“弯路”：在非约束生成器映射中，两个相似图像之间的潜在空间插值可以通过 RGB 空间中的任意数量的远距离图像。通过正则化，鼓励映射将潜在的图像放置在潜在空间的不同区域中，以便在任意两个端点之间获得较短的图像路径。

D. 投影方式细节

给定目标图像 x，我们试图找到相应的 $\mathbf{w} \in \mathcal{W}$ 和表示为 $\boldsymbol{n}_i \in \mathbb{R}^{r_i \times r_i}$ 的每层噪声图，其中 i 是层索引，并且 r_i 表示第 i 个噪声图的分辨率。 分辨率为 1024×1024 的基线 StyleGAN 生成器具有 18 个噪声输入，即每个分辨率为 4×4 到 1024×1024 像素的噪声输入为 2 个。改进后的架构减少了一个噪声输入，因为我们没有将噪声添加到学习到 4×4 常数中（图 2）。

在优化之前，我们通过在映射网络 f 上运行 10000 个随机潜码 z 来计算 $\boldsymbol{\mu}_\mathbf{w} = \mathbb{E}_\mathbf{z} f(\mathbf{z})$。我们还通过计算 $\sigma_\mathbf{w}^2 = \mathbb{E}_\mathbf{z} \|f(\mathbf{z}) - \boldsymbol{\mu}_\mathbf{w}\|_2^2$，即到中心的平均欧几里德距离，来近似 W 的尺度。

在优化开始时，我们对所有 i 初始化 $\mathbf{w} = \boldsymbol{\mu}_\mathbf{w}$ 和 $\boldsymbol{n}_i = \mathcal{N}(\mathbf{0}, \mathbf{I})$。训练参数为

w 的分量以及所有噪声映射$\boldsymbol{n}_i$中的所有分量。使用带有默认参数的 Adam 优化器[25]可以进行 1000 次迭代的优化。最大学习速率为$\lambda_{max} = 0.1$，在前 50 次迭代中，线性学习从零线性增加，在最近 250 次迭代中，使用余弦进度表，线性学习降低到零。在优化的前四分之三中，当将损失函数评估为$\tilde{\mathbf{w}} = \mathbf{w} + \mathcal{N}(0, 0.05\,\sigma_{\mathbf{w}} t^2)$时，我们在 w 上添加高斯噪声，其中在前 750 次迭代中 t 从 1 变为零。这增加了优化的随机性，并稳定了全局最优的发现。

鉴于我们正在明确优化噪声图，因此必须小心避免优化将实际信号潜入噪声图。因此，除了图像质量项外，我们在损失函数中还包括几个噪声图正则项。图像质量项是目标图像 x 与合成图像之间的 LPIPS [49] 距离：$L_{image} = D_{\text{LPIPS}}[\boldsymbol{x}, g(\tilde{\mathbf{w}}, \boldsymbol{n}_0, \boldsymbol{n}_1, \ldots)]$。为了提高性能和稳定性，我们在计算 LPIPS 距离之前将这两个图像下采样到 256×256 分辨率。噪声图的正则化是在多个分辨率范围内执行的。为此，我们通过对 2×2 个像素邻域求平均，并在每一步乘以 2 来保留预期的单位方差，从而为每个大于 8×8 的噪声图形成低至 8×8 分辨率的金字塔。这些下采样后的噪声图仅用于正则化，不参与合成。

让我们用$\boldsymbol{n}_{i,0} = \boldsymbol{n}_i$表示原始噪声图，通过$\boldsymbol{n}_{i,j>0}$表示降采样的版本。类似地，令$r_{i,j}$为原始（j = 0）或下采样（j> 0）噪声图的分辨率，以便$r_{i,j+1} = r_{i,j}/2$。噪声图$\boldsymbol{n}_{i,j}$的正则项为

$$L_{i,j} = \left(\frac{1}{r_{i,j}^2} \cdot \sum_{x,y} \boldsymbol{n}_{i,j}(x,y) \cdot \boldsymbol{n}_{i,j}(x-1,y) \right)^2 + \left(\frac{1}{r_{i,j}^2} \cdot \sum_{x,y} \boldsymbol{n}_{i,j}(x,y) \cdot \boldsymbol{n}_{i,j}(x,y-1) \right)^2,$$

噪声图被认为包裹在边缘。因此，正则化项是在一个像素水平和垂直移动时，分辨率归一化自相关系数的平方和，对于正态分布的信号，其应该为零。那么，总损失项为$L_{total} = L_{image} + \alpha \sum_{i,j} L_{i,j}$。在所有测试中，我们都使用了噪声调整权重$\alpha = 10^5$。此外，在每个优化步骤之后，我们将所有噪声图归一化为零均值和单位方差。图 18 说明了噪声正则化对所得噪声图的影响。

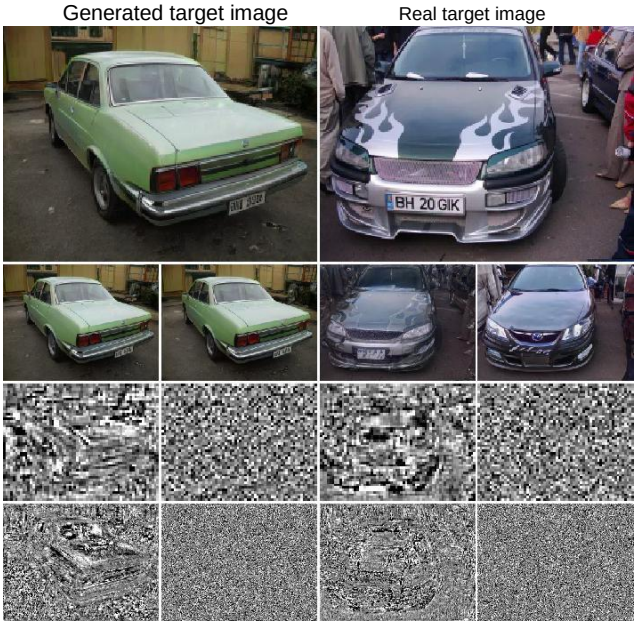


图 18.噪声正则化在潜在空间投影中的效果，在此我们还优化了合成网络的噪声输入的内容。从上到下：目标图像，重新合成的图像，两个分辨率不同的噪声图的内容。当在此测试中关闭正则化时，我们仅将噪声图标准化为零均值和单位方差，从而导致优化将信号潜入噪声图。启用噪声调整可防止这种情况。此处使用的模型对应于表 1 中的配置 F。