

生成对抗网络调研报告

杨逸文



北京知音智慧科技有限公司 AI 实验室

2018 年 5 月 18 日

目录

1 GAN 背景介绍.....	6.
1.1 人工智能.....	6.
1.2 生成模型.....	7.
1.2.1 生成模型的意义	7.
1.2.2 GAN 的由来	7.
1.2.2.1 神经网络的深化.....	7.
1.2.2.2 对抗思想的产生.....	8.
1.2.2.3 GAN 的诞生	8.
2 GAN 的基本理论与实现模型.....	8.
2.1 GAN 基本概念.....	8.
2.1.1 学习数据分布	8.
2.1.2 标记符号	9.
2.2 GAN 的基本原理.....	9.
2.3 GAN 生成器与判别器的损失函数及其训练算法.....	9.
2.4 存在的问题.....	10.
2.4.1 训练速度	10.
2.4.2 梯度消失	11.
2.5 个人观点	11.
3. GAN 的衍生模型.....	12.
3.1 DCGAN.....	13.
3.1.1 DCGAN 模型结构	13.
3.1.2 DCGAN 实验应用	14.
3.1.2.1 DCGAN 生成图片	14.
3.1.2.2 DCGAN 隐空间的应用.....	15.
3.1.2.3 DCGAN 控制生成的物体.....	16.
3.1.2.4 DCGAN 向量计算	17.

3.1.3 个人观点	18.
3.2. WGAN	19.
3.2.1 Earth-Mover (Wasserstein) 距离	19.
3.2.2 WGAN 生成器与判别器的损失函数及其训练算法	20.
3.2.3 WGAN 算法的局限性	21.
3.2.4 WGAN 的实验应用	21.
3.2.4.1 Wasserstein 距离的优势	21.
3.2.4.2 WGAN 生成图片的优势	21.
3.3 WGAN-GP	22.
3.3.1 WGAN-GP 生成器与判别器的损失函数及其训练算法	22.
3.3.2 WGAN-GP 的贡献	24.
3.3.3 WGAN-GP 的实验应用	24.
3.4 C-GAN	25.
3.4.1 CGAN 生成器与判别器的损失函数及其训练算法	26.
3.4.2 CGAN 的网络结构	26.
3.4.3 CGAN 的实验应用	26.
3.4.3.1 CGAN 在 MNIST 数据集上的应用	26.
3.4.3.2 CGAN 图像自动标注	27.
3.4.4 未来的工作	27.
3.4.5 个人观点	28.
3.5 LAPGAN (拉普拉斯金字塔生成式对抗网络)	28.
3.5.1 拉普拉斯金字塔	29.
3.5.2 LAPGAN 生成图片的过程	29.
3.5.3 LAPGAN 的博弈过程 (训练过程)	29.
3.5.4 LAPGAN 生成器与判别器的损失函数及其训练算法	30.
3.5.5 LAPGAN 的实验应用	30.
3.5.6 个人观点	30.
3.6 Info-GAN	32.

3.6.1 互信息.....	32.
3.6.2 InfoGAN 生成器与判别器的损失函数及其训练算法	32.
3.6.2.1 $I(c; G(z, c))$ 的计算	33.
3.6.3 InfoGAN 的实验应用	33.
3.7 SeqGAN	35.
3.7.1 SeqGAN 生成器与判别器的损失函数及其训练算法	35.
3.7.2 SeqGAN 的实验应用	37.
3.8 LSGAN、Semi-GAN、BiGANs、ACGAN.....	37.
4. GAN 算法的比较.....	38.
5. GAN 的应用领域	38.
5.1 图像和视觉领域	38.
5.2 语音和语言领域	39.
5.2.1 个人观点.....	40.
5.3 其他领域	40.
6. GAN 的缺陷和发展趋势	40.
7. 参考文献	41.

生成对抗学习综述

杨逸文¹ 赵亚伟^{1,2}

(1. 北京知音智慧科技有限公司 AI 实验室, 北京 100028)

(2. 中国科学院大学 大数据分析技术实验室, 北京 100049)

摘要：生成对抗网络 (GAN) 提供了一种利用深度学习的方法来学习没有标签或者标签很少的数据。它们利用博弈论的思想，用一对儿网络相互对抗来生成以假乱真的数据。对抗网络的提出促进了多种多样的应用，包括图像生成，语义图像的编辑，风格转移，超分辨率图像的生成以及分类等等。不仅包括生成数据方面的应用，利用判别器判别物体的真假可能也是未来的一个应用。这篇论文的目的提供了 GAN 的综述，包括 GAN 的诞生背景，经典类型 GAN 的详细介绍 (背景介绍，损失函数，训练算法，实验应用，存在的缺陷分析，个人看法等)，GAN 算法的相互比较，GAN 的应用领域，GAN 的缺陷和发展趋势。

关键词 生成对抗网络，人工智能，生成式模型。

1. GAN 背景介绍

1.1 人工智能

近年来,随着计算能力的提高和各行业数据量的剧增,人工智能取得了快速发展,使得研究者对人工能的关注度和社会大众对人工智能的憧憬空前提升^[1]。学术界普遍认为人工智能分为两个阶段^[2]:感知阶段和认知阶段。在感知阶段,机器能够接收来自外界的各种信号,例如视觉信号、听觉信号等,并对此作出判断,对应的研究领域有图像识别语音识等。在认知阶段,机器能够对世界的本质有一定的理解,不再是单纯机械地做出判断。然而,理解无论对人类还是人工智能都是内在的表现,无法直接测量,只能间接从其他方面推测。如何衡量人工智能的理解程度,虽然没有定论,但是著名学者Feynman 有句名言 “What I cannot create, I do not understand.” (不可造者,未能知也。)” 这说明机器制造事物的能力从某种程度上取决于机器对事物的理解。而GAN 作为典型的生成式模型,其生成器具有生成数据样本的能力。这种能力在一定程度上反映了它对事物的理解。因此,GAN 有望加深人工智能的理解层面的研究。

1.2 生成模型

生成式模型不仅在人工智能领域占有重要地位,生成方法本身也具有很大的研究价值。生成方法和判别方法是机器学习中监督学习方法的两个分支。生成式模型是生成方法学习得到的模型。生成方法涉对数据的分布假设和分布参数学习,并能够根据学习而来的模型采样出新的样本。生成式模型从研究出发点的角度可以分为两类^[2]:人类理解数据的角度和机器理解数据的角度。

从人类理解数据的角度出发,典型的做法是先对数据的显式变量或者隐含变量进行分布假设,然后利用真实数据对分布的参数或包含分布的模型进行拟合或训练,最后利用学习到的分布或模型生成新的样本。这类生成式模型涉及的主要方法有最大似然估计法,近似法^[3,4]、马尔科夫链方法^[5,6]等。从这个角度学习到的模型具有人类能够理解的分布,但是对机器学习来说具有不同的限制。例如,以真实样本进行最大似然估计,参数更新直接来自于数据样本,导致学习到的生成式模型受到限制。而采用近似法学习到的生成式模型由于目标函数难解一般只能在学习过程中逼近目标函数的下界,并不是直接对目标函数的逼近。马尔科夫链方法既可以用于生成式模型的训练又可以用于新样本的生成,但是马尔科夫链的计算复杂度较高。

从机器理解数据的角度出发,建立的生成式模型一般不直接估计或拟合分布,而是从未明确假设的分布中获取采样的数据^[7],通过这些数据对模型进行修正。这样得到的生成式模

型对人类来说缺乏可解释性,但是生成的样本却是人类可以理解的。以此推测,机器以人类无法显式理解的方式理解了数据并且生成了人类能够理解的新数据。在GAN提出之前,这种从机器理解数据的角度建立的生成式模型一般需要使用马尔科夫链进行模型训练,效率较低,一定程度上限制了其系统应用。而GAN由此而生,新的生成模式突破了已有的障碍,为实现人工智能的四个层次奠定了基础。

1.2.1 生成模型的意义

总体来说,生成模型的研究价值主要基于以下几个方面^[8]:

1. 从生成模型中训练和采样数据能很好的测试我们表示和操作高维概率分布的能力。而这种能力在数学和工程方面都有广泛的应用。
2. 生成模型可以通过很多方式应用到强化学习中。强化学习算法一般分为两类:基于模型的(包含生成模型)和与模型无关的。时间序列生成模型可以模拟将来的很多种可能。例如,当我们学习一个条件概率分布模型时,当前时刻的状态和假设的动作作为输入,用生成模型来产生未来的状态。
3. 生成模型还可以填补缺失信息。在半监督学习中的应用,由于真实环境下,很少有带标注的信息,而且标注信息往往依靠高成本的人工标注。生成模型,特别是 GAN,能够在很少标注样本的情况下,提高半监督学习算法的泛化能力。又比如为了提高图像的分辨率,需要给低分辨率的图像填补缺失信息来合成高分辨率图像。相对于 GAN,其他生成模型往往对多种可能的生成图像进行平均,从而造成图像模糊。而 GAN 则会生成多个可能的清晰图像。

1.2.2 GAN 的由来

在GAN提出之前,这种从机器理解数据的角度建立的生成式模型一般需要使用马尔科夫链进行模型训练,效率较低,一定程度上限制了其系统应用。要真正实现人工智能的四个层次,就需要设计新的生成式模型来突破已有的障碍^[2]。

1.2.2.1 神经网络的深化

过去10年来,随着深度学习技术^[9,10]在各个领域取得巨大成功,神经网络研究再度崛起。神经网络作为深度学习的模型结构,得益于计算能力的提升和数据量的增大,一定程度上解决了自身参数多、训练难的问题,被广泛应用于解决各类问题中。例如,深度学习技术在图像分类问题上取得了突破性的效果^[11,12],显著提高了语音识别的准确率^[13],又被成功应用于自然语言理解领域^[14]。神经网络取得的成功和模型自身的特点是密不可分的。在训练方面,神经网络能够采用通用的反向传播算法,训练过程容易实现;在结构方面,神经网络的结构设计自由灵活,局限性小;在建模能力方面,神经网络理论上能够逼近任意函数,应

用范围广。另外，计算能力的提升使得神经网络能够更快地训练更多的参数，进一步推动了神经网络的流行。

1.2.2.2 对抗思想的产生

从机器学习到人工智能，对抗思想被成功引入若干领域并发挥作用。博弈、竞争中均包含着对抗的思想。博弈机器学习^[15]将博弈论的思想与机器学习结合，对人的动态策略以博弈论的方法进行建模，优化广告竞价机制，并在实验中证明了该方法的有效性。围棋程序AlphaGo^[16]战胜人类选手引起大众对人工智能的兴趣，而AlphaGo的中级版本在训练策略网络的过程中就采取了两个网络左右互搏的方式，获得棋局状态、策略和对应回报，并以包含博弈回报的期望函数作为最大化目标。在神经网络的研究中曾有研究者利用两个神经网络互相竞争的方式对网络进行训练^[17]，鼓励网络的隐层节点之间在统计上独立，将此作为训练过程中的正则因素。还有研究者采用对抗思想来训练领域适应的神经网络^[18,19]：特征生成器将源领域数据和目标领域数据变换为高层抽象特征，尽可能使特征的产生领域难以判别；领域判别器基于变换后的特征，尽可能准确地判别特征的领域。对抗思想应用于机器学习或人工智能取得的诸多成果，也激发了更多的研究者对GAN 的不断挖掘。

1.2.2.3 GAN 的诞生

GAN的核心思想来源于博弈论的纳什均衡。它设定参与游戏双方分别为一个生成器(Generator)和一个判别器(Discriminator)，生成器的目的是尽量去学习真实的数据分布，而判别器的目的是尽量正确判别输入数据是来自真实数据还是来自生成器；为了取得游戏胜利，这两个游戏参与者需要不断优化，各自提高自己的生成能力和判别能力，这个学习优化过程就是寻找二者之间的一个纳什均衡。为了学习优化，我们设定目标函数并设计训练过程，为了实现目标函数和训练过程，我们设计神经网络的架构。所以，笔者认为，目标函数和神经网络的架构是GAN核心的两个部分。

2. GAN 的理论与实现模型

2.1 GAN 基本概念

2.1.1 学习数据分布

信号处理与统计的核心问题是密度估值：从显示或者隐式、系数或者非系数方面获得真实数据的表示。这是 GAN 的核心动机。在 GAN 的文献中，数据分布一般是指数据的概率密度或者概率质量函数。GAN 的目的则是学习训练数据的分布，生成满足训练数据分布的逼真数据。生成样本的任务可能是下游任务，比如语义图片编辑，数据扩充，风格转移。也可能是分类和图像提取^[20]。

GAN 的判别器和生成器的结构采取神经网络的结构，结构一般采用全连接层或者卷积网络。所有网络的训练采用BP^[20]方式。

2.1.2 标记符号

GAN 生成器一般采用随机多维变量 Z 为输入变量。 $P_{data}(x)$ 做为训练数据的概率密度函数。 $P_g(x)$ 则作为生成器生成数据的概率密度函数。 G 与 D 分别代表生成器和判别器。生成器和判别器的网络具有一套系数(权值)分别表示为 θ_D 和 θ_G 。这套权值在训练过程中不断优化^[20]。

2.2 GAN 的基本原理

GAN 的计算流程与结构如图1所示。任意可微分的函数都可以用来表示GAN的生成器和判别器,由此,我们用可微分函数 D 和 G 来分别表示判别器和生成器,它们的输入分别为真实数据 x 和随机变量 z 。 $G(z)$ 则为由 G 生成的尽量服从真实数据分布 P_{data} 的样本。如果判别器的输入来自真实数据,标注为1。如果输入样本为 $G(z)$,标注为0. 这里 D 的目标是实现对数据来源的二分类判别: 真(来源于真实数据 x 的分布)或者伪(来源于生成器的伪数据 $G(z)$),而 G 的目标是使自己生成的伪数据 $G(z)$ 在 D 上的表现 $D(G(z))$ 和真实数据 x 在 D 上的表现 $D(x)$ 一致,这两个相互对抗并迭代优化的过程使得 D 和 G 的性能不断提升,当最终 D 的判别能力提升到一定程度,并且无法正确判别数据来源时,可以认为这个生成器 G 已经学到了真实数据的分布。

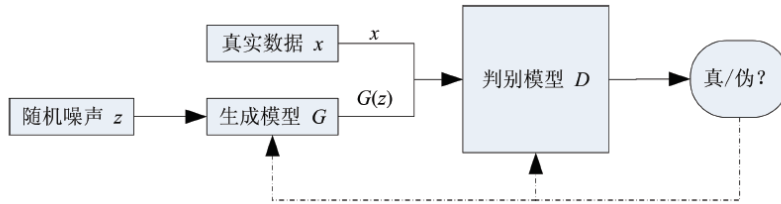


图1. GAN的计算流程与结构图

2.3 生成器与判别器的损失函数及其训练算法

GAN 的优化问题是一个极小极大化问题，GAN 的目标函数可以描述如下：

$$\min_G \max_D E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (1)$$

其中, x 采样于真实数据分布 $P_{data}(x)$, z 采样于先验分布 $p_z(z)$ 例如高斯噪声分布), $E(\cdot)$ 表示计算期望值。给定生成器 G , 我们需要最小化式 (2) 来得到最优解,

$$Obj^D(\theta_D, \theta_G) = -\frac{1}{2} E_{x \sim p_{data}(x)} [\log D(x)] - \frac{1}{2} E_{z \sim p_z(z)} [\log(1 - D(g(z)))] \quad (2)$$

在连续空间上, 根据期望的性质和积分的意义可以将 (2) 写为以下形式:

$$\begin{aligned} Obj^D(\theta_D, \theta_G) &= -\frac{1}{2} \int_x p_{data}(x) \log(D(x)) dx - \frac{1}{2} \int_z p_z(z) \log(1 - D(g(z))) dz \\ &= -\frac{1}{2} \int_x [p_{data}(x) \log(D(x)) + p_g(x) \log(1 - D(x))] dx \end{aligned} \quad (3)$$

对任意的非零实数 m 和 n , 且实数值 $y \in [0,1]$, 表达式

$$-m \log(y) - n \log(1-y) \quad (4)$$

在 $\frac{m}{m+n}$ 处得到最小值。因此, 给定生成器 G 的情况下, 目标函数(3)在

$$D_G^*(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \quad (5)$$

处取得的最小值, 而生成器 G 的损失函数为

$$\begin{aligned} \text{Obj}^G(\theta_G) &= -\text{Obj}^D(\theta_D, \theta_G) \\ &= E_{x \sim p_g(x)} \log(1 - D(G(x))) \end{aligned} \quad (6)$$

总之, 对于GAN的学习过程, 我们需要训练模型 D 来最小化(2)式, 同时, 我们需要训练模型 G 来最小化(6)式。这里可以采用交替优化的方法: 先固定生成器 G , 优化判别器 D , 使得 D 的判别准确率最大化; 然后固定判别器 D , 优化生成器 G , 使得 D 的判别准确率最小化。当且仅当 $p_{data} = p_g$ 时达到全局最优解。训练GAN时, 同一轮参数更新中, 一般对 D 的参数更新 k 次再对 G 的参数更新1次。算法如图2所示。

2.4 存在的问题

2.4.1 训练速度

对于生成器 G 的损失函数 $\log(1 - D(x))$, 在 $D(x)$ 接近于0的时候, 函数十分平滑, 梯度非常小。如图3所示。这就会导致在训练的初期, G 想要骗过 D , 变化缓慢, 而 $-\log(D(x))$ 函数的趋势和 G 的损失函数是一样的, 都是递减的。但是它的优势是在 $D(x)$ 接近0的时候, 梯度很大, 有利于训练, 在 $D(x)$ 越来越大之后, 梯度减小。这也符合实际情况, 在

```

for number of training iterations do
  for  $k$  steps do
    • Sample minibatch of  $m$  noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from noise prior  $p_g(z)$ .
    • Sample minibatch of  $m$  examples  $\{x^{(1)}, \dots, x^{(m)}\}$  from data generating distribution  $p_{data}(x)$ .
    • Update the discriminator by ascending its stochastic gradient:
      
$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[ \log D(x^{(i)}) + \log (1 - D(G(z^{(i)}))) \right].$$

    end for
    • Sample minibatch of  $m$  noise samples  $\{z^{(1)}, \dots, z^{(m)}\}$  from noise prior  $p_g(z)$ .
    • Update the generator by descending its stochastic gradient:
      
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(z^{(i)}))).$$

  end for
  The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

```

图2 GAN 训练算法

初期训练速度快，到后期速度减慢。所以原作者 Goodfellow 将损失函数变为 $-\log(D(x))^{[21]}$ ，这样提高了训练速度，训练的准度也随之提高了。

2.4.2 梯度消失

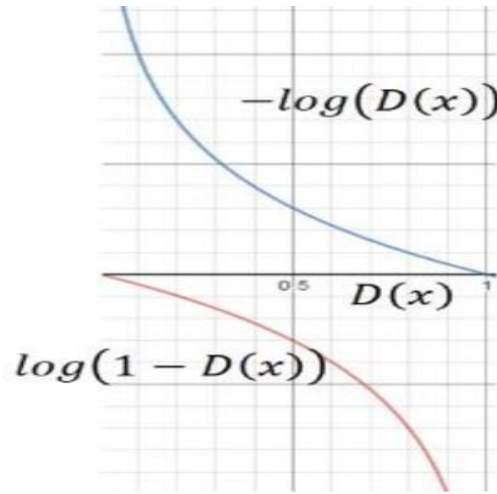
经过实验发现，经过许多次训练，损失函数一直都是常数。有一种解释证明了在 $D(x)$ 达到最优值时，

$$\min_G \max_D E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))] \approx \text{JSD}(p_{data}(x) || p_g(x)) \quad (7)$$

JSD散度用来计算两个数据分布之间的差异性。作者从而从JSD散度分析真实数据和生成数据分布之间的差异，JSD为常数时，存在三个情况：

- (1) $p_{data}(x)$ 与 $p_g(x)$ 都为0
- (2) $p_{data}(x)=0$
- (3) $p_g(x)=0$

这三种情况直接说明了 $p_{data}(x)$ 与 $p_g(x)$ 没有交集。但是实际上两个分布是有交集的，造成这个的原因是损失函数并没有像上文推倒的那样真正用积分计算期望值，而是使用均值的方法。然而，如果两个分布映射在高维空间上都很窄的话，两个分布可能确实没交集或者交集很小。



图三 GAN 生成器损失函数

2.5. 个人观点

1. 既然梯度消失的原因是损失函数并没有像上文推倒的那样真正用积分计算期望值，而是使用均值的方法，那不妨就让损失函数用积分计算期望。所以判别器和生成器的损失函数的训练方式可以改为如下形式：

$$\nabla_{\theta_d} [\int_x p_{data}(x) \log(D(x)) dx - \int_z p_z(z) \log(1 - D(g(z))) dz] \quad (8)$$

$$\nabla_{\theta_g} [\int_z p_z(z) \log(1 - D(g(z))) dz] \quad (9)$$

在这里，假设 $p_{data}(x)$ 与 $p_z(z)$ 服从高斯分布。

2. 为了避免两个数据分布映射在高维空间上都很窄的问题，最简单地实验方法就是增加 z 的维数来增加数据分布的交集。

3. 如 7 式所说，当损失函数解释为两个数据分布的 JSD 散度时，即使两个数据分布高度重合，在高维空间里，也会有很多空间是两个数据分布独有的，在这种情况下，损失函数又变为了常数，我们又要面临梯度消失的问题。所以无论两个数据分布不相交、相交、高度相交，GAN 的损失函数都会面临梯度消失的问题。实际的损失函数的图像如果用二维空间表示的话，会出现平滑的线段，并不是像图三那样简单。这也许就是 GAN 损失函数存在的问题。

3. GAN 的衍生模型

自 Goodfellow 等于 2014 年提出 GAN 以来，各种基于 GAN 的衍生模型被提出，这些模型的创新点包括模型结构改进、理论扩展及应用等。部分衍生模型的计算流程与结构以及损失函数如图 4，图 5 所示。

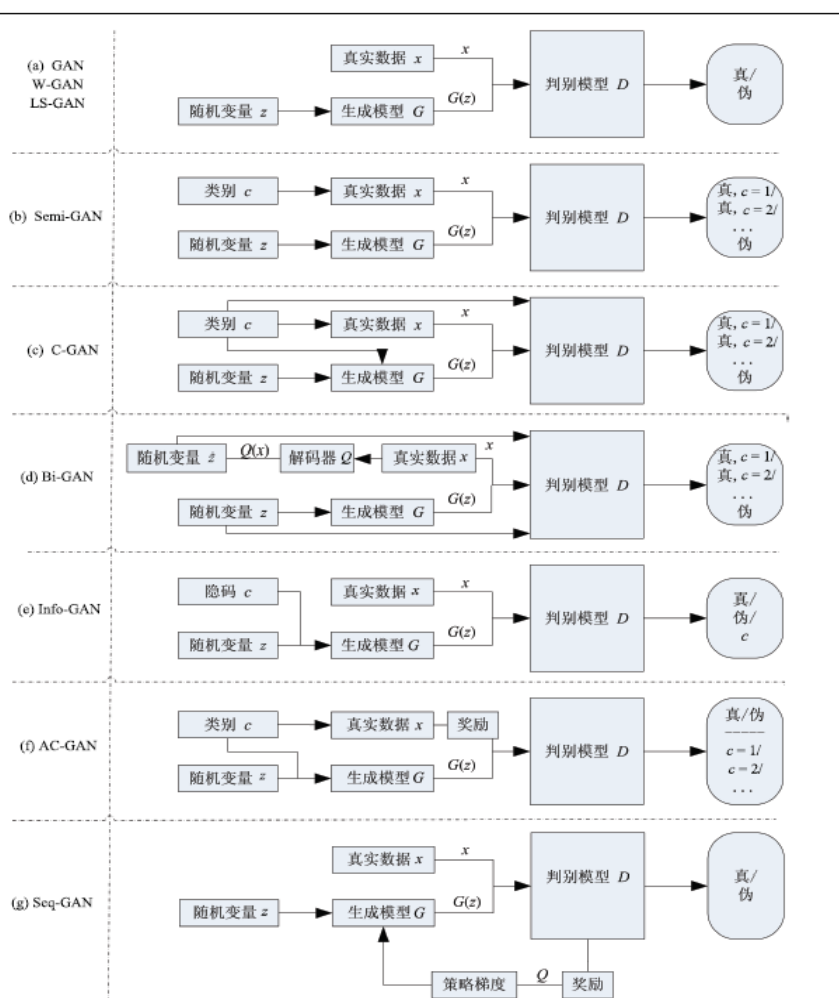


图4 衍生模型的计算流程及结构

GAN	DISCRIMINATOR LOSS	GENERATOR LOSS
MM GAN	$\mathcal{L}_D^{\text{GAN}} = -\mathbb{E}_{x \sim p_d} [\log(D(x))] - \mathbb{E}_{\hat{x} \sim p_g} [\log(1 - D(\hat{x}))]$	$\mathcal{L}_G^{\text{GAN}} = \mathbb{E}_{\hat{x} \sim p_g} [\log(1 - D(\hat{x}))]$
NS GAN	$\mathcal{L}_D^{\text{NSGAN}} = -\mathbb{E}_{x \sim p_d} [\log(D(x))] - \mathbb{E}_{\hat{x} \sim p_g} [\log(1 - D(\hat{x}))]$	$\mathcal{L}_G^{\text{NSGAN}} = -\mathbb{E}_{\hat{x} \sim p_g} [\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_D^{\text{WGAN}} = -\mathbb{E}_{x \sim p_d} [D(x)] + \mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})]$	$\mathcal{L}_G^{\text{WGAN}} = -\mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})]$
WGAN GP	$\mathcal{L}_D^{\text{WGANP}} = \mathcal{L}_D^{\text{WGAN}} + \lambda \mathbb{E}_{\hat{x} \sim p_g} [(\nabla D(\alpha x + (1 - \alpha \hat{x})) _2 - 1)^2]$	$\mathcal{L}_G^{\text{WGANP}} = -\mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})]$
LS GAN	$\mathcal{L}_D^{\text{LSGAN}} = -\mathbb{E}_{x \sim p_d} [(D(x) - 1)^2] + \mathbb{E}_{\hat{x} \sim p_g} [D(\hat{x})^2]$	$\mathcal{L}_G^{\text{LSGAN}} = -\mathbb{E}_{\hat{x} \sim p_g} [(D(\hat{x}) - 1)^2]$
DRAGAN	$\mathcal{L}_D^{\text{DRAGAN}} = \mathcal{L}_D^{\text{GAN}} + \lambda \mathbb{E}_{\hat{x} \sim p_d + \mathcal{N}(0, c)} [(\nabla D(\hat{x})) _2 - 1)^2]$	$\mathcal{L}_G^{\text{DRAGAN}} = \mathbb{E}_{\hat{x} \sim p_g} [\log(1 - D(\hat{x}))]$
BEGAN	$\mathcal{L}_D^{\text{BEGAN}} = \mathbb{E}_{x \sim p_d} [x - \text{AE}(x) _1] - k_t \mathbb{E}_{\hat{x} \sim p_g} [\hat{x} - \text{AE}(\hat{x}) _1]$	$\mathcal{L}_G^{\text{BEGAN}} = \mathbb{E}_{\hat{x} \sim p_g} [\hat{x} - \text{AE}(\hat{x}) _1]$

图5. GANs损失函数

3.1 DCGAN

在GAN的影响下，Alec^[22]等人提出了卷积网络架构，这极大地促进了图片生成的研究。总之，作者在DCGAN研究上做出了以下几点贡献：

1. 作者等人提出并评估了一套利用卷积网络的架构使之可以稳定地训练，作者将它命名为DCGAN。
2. 作者等人用训练好的判别器 D 来进行图片分类的任务，结果表明，模型 D 要优于其他多种非监督的算法。
3. 实验证明，学到的具体 filter 可以生成具体的物体（图片）。
4. 生成器 G 可以生成具有语义效果的样本图片。

3.1.1 DCGAN 模型结构

DCGAN 的网络结构在卷积网络的基础上做了如下几点变化：

1. 无论是生成器和判别器，池化层都被卷积层取代。也就是说，生成器上采用反卷积并用反卷积替代池化层，所以这个过程是由低维数据生成高维数据的过程。而判别器则采用卷积并用卷积替代池化层，所以这个过程是由高维数据生成低维数据的过程。
2. 消除全连接层。这是一个趋势。最典型的例子Mordvintsev^[23]等人采用全局平均池化进行图像分类。但是作者发现使用全局平均池化虽然增加了模型的稳定性（防止过拟合），但是却影响了收敛的速度。所以作者直接采用卷积层连接输入与输出，实验表明良好。下面给出生成器的网络结构图，如图 6 所示：

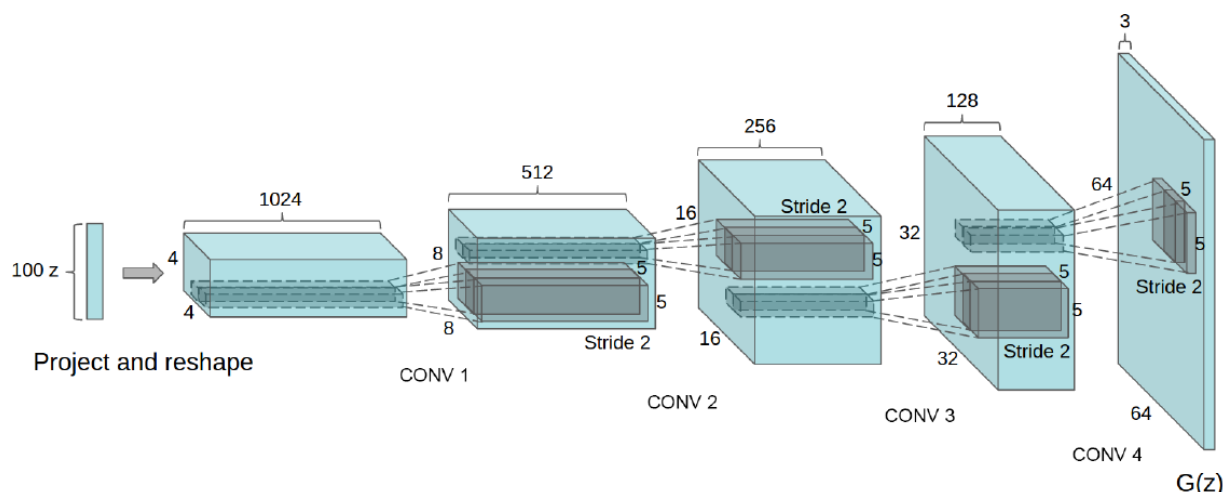


图 6：使用 LSUN 图像数据集的生成器模型结构。无池化层，无全连接层。

3. 采用批次标准化（batch normalization）^[24]：首先，它使输入的平均值为 1，方差为 0，有利于初始化的计算和在深度模型里面的梯度计算。这也防止了生成器将所有的样本都收敛到同一点上，这在原始的 GAN 上普遍发生。然而，将批次标准化用在所有的网络层上会导致样本震荡和模型不稳定性，所以作者没有将批次标准化应用在生成器的输出层和判别器的输入层。

4. 在生成器上，输出层的激活函数采用 Tanh 函数，其它层采用 ReLU 激活函数。作者发现这样做可以使模型学的更快以及捕捉到训练数据分布的色彩空间。在判别器上，作者发现 leaky rectified 激活函数效果良好，特别是处理高像素的问题。而原始的 GAN 采用的是 maxout 激活函数。

3.1.2 DCGAN 实验应用

作者在论文里给出了详细的实验过程，实验结果，评判标准。下面我们来看看 DCGAN 的能力。

3.1.2.1 DCGAN 生成图片

图 7 图 8 分别是 DCGAN 在 LSUN 图片经过一次训练和五次训练得到的效果。可以看到 5 次的效果要比一次的好。

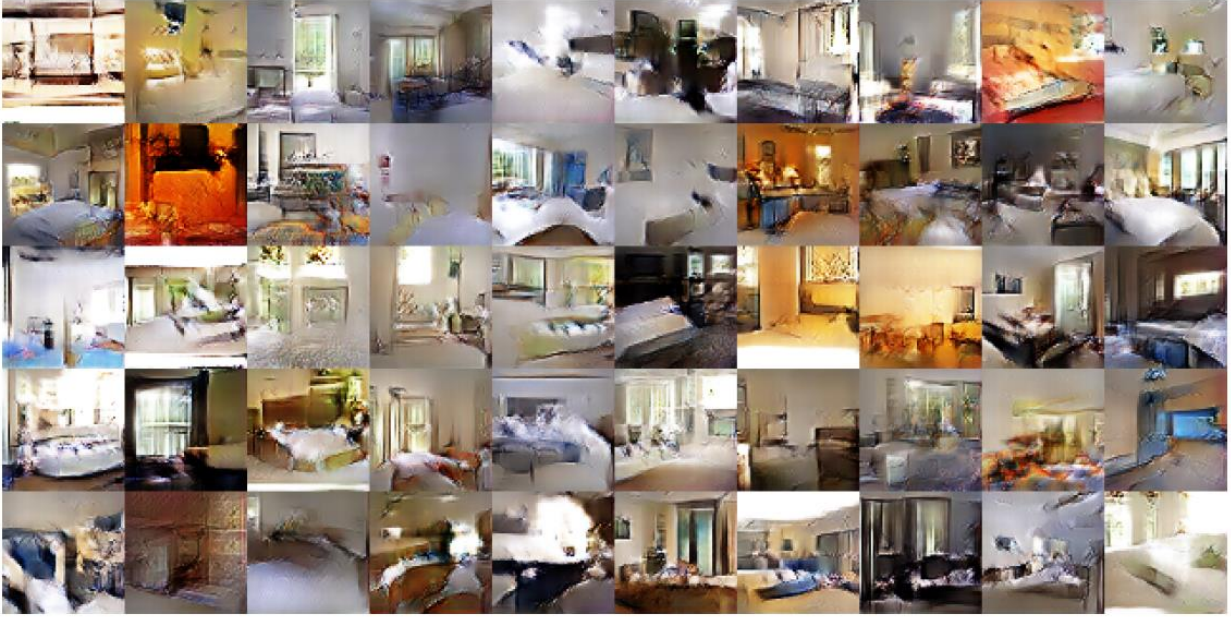


图7：DCGAN经过一次训练生成的图片

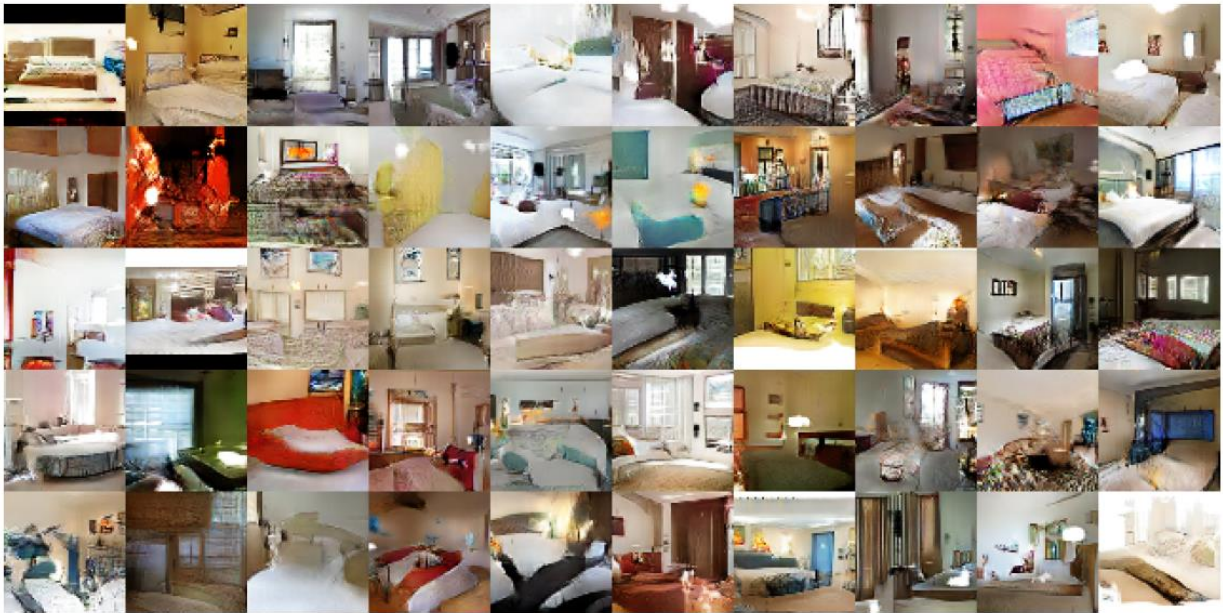


图8：DCGAN经过五次训练生成的图片

3.1.2.2 DCGAN 隐空间的应用

通过慢慢的调整初始向量来探索隐空间 z 是如何影响最终图片的生成的。这样，既可以探索图片特征是如何折叠到隐空间的，又可以判断这些图片是由于真正学习到了语义特征还是只是记住了图片，如图 9 所示。

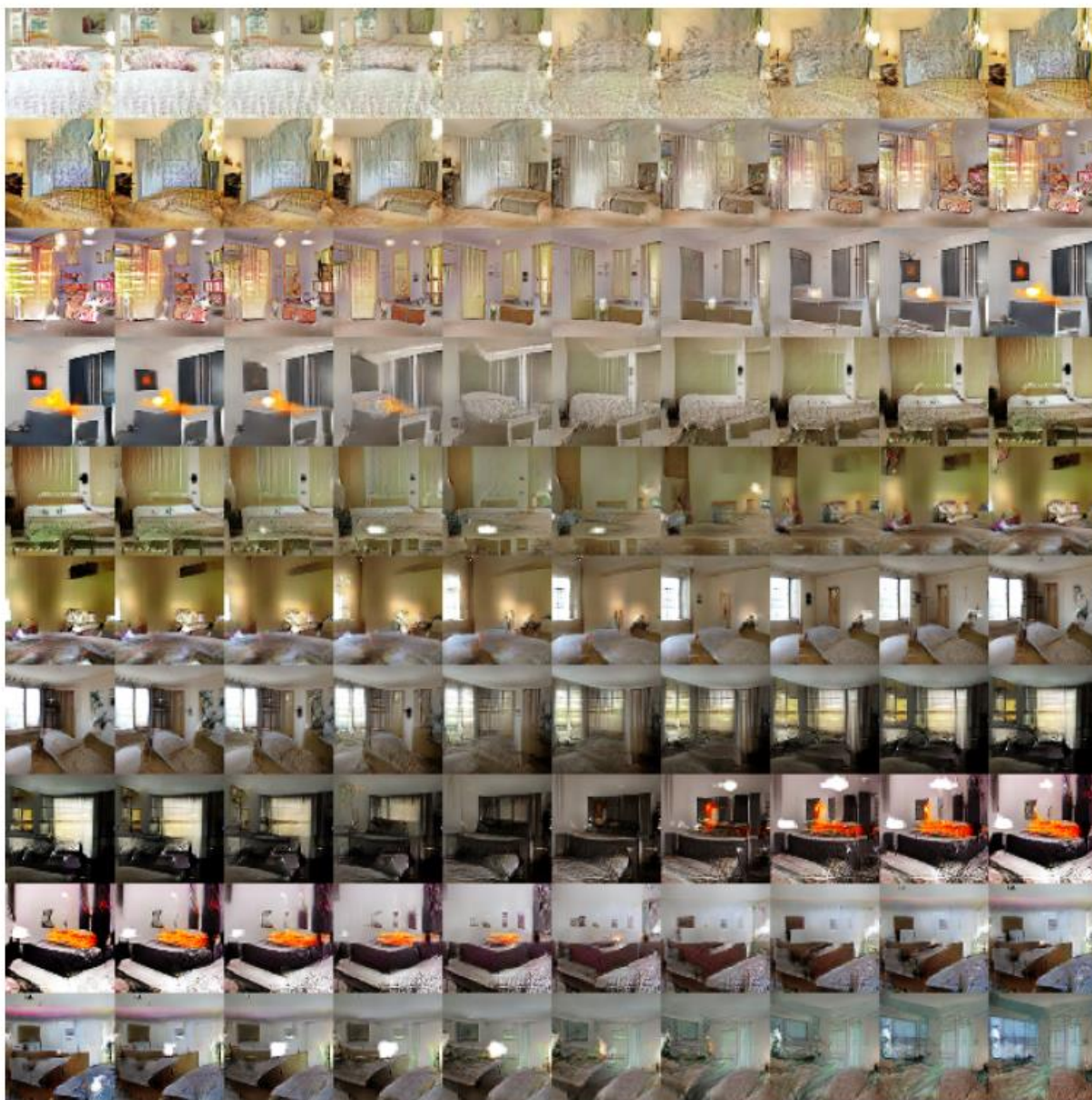


图9：通过变化隐藏空间 z 生成的图片

我们可以看到，第6行逐渐有了窗户，而第四行的电视逐渐消失。从中可以证明DCGAN真正学到了图片的分布而不是单纯地复制图片。

3.1.2.3 DCGAN 控制生成的物体（这个吊）

在卷积层的 filter 上，假设知道哪些 filter 控制着图片里的某个物体，那么将这些 filter 移除，从而删除掉图片里的这个物体。如图 10 所示：



图 10: DCGAN 控制生成图片里的具体物体。上层表示模型未经修改生成的图片，下层表示模型移除了控制窗户的 filters 而生成的图片。可以看到在下层图片里，有一些丢失了窗户，有一些将窗户变成了形状类似的物体（门和窗户）。虽然下层图片的视觉效果变差了，但整体的场景效果还是可以接受的。未来的实验可以尝试移除其他的物体或者修改物体。

3. 1. 2. 4 DCGAN 向量计算

不仅如此，DCGAN 的功能还能生成具有语义关系的图片，如图 11 所示：

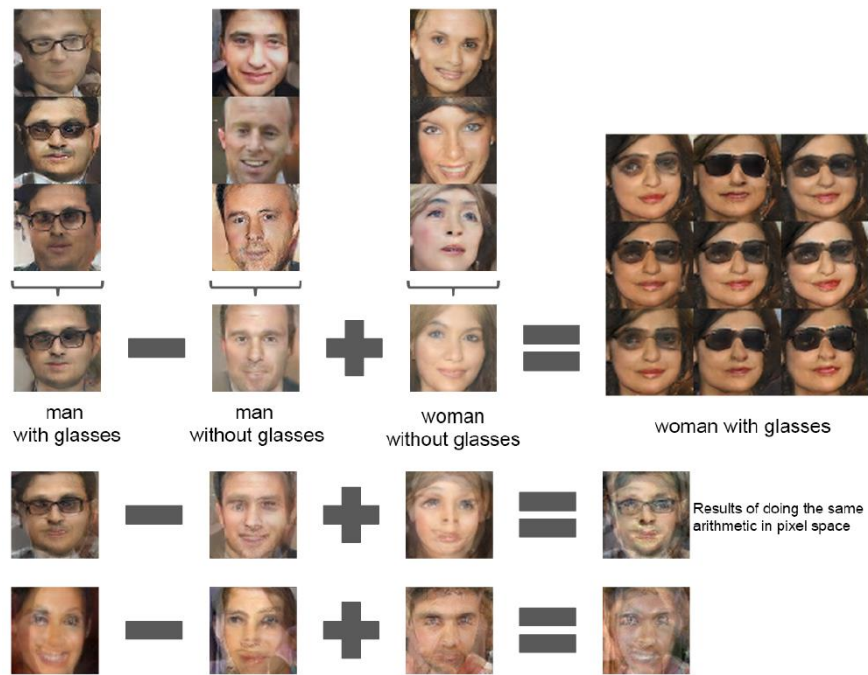


图11. DCGAN生成具有语义关系的图片。每一列的最下面表示的是潜在变量 z 的平局值生成的图片。使用单张图片的表示并不稳定，所有使用三张图片的平均值来进行计算。

并且DCGAN可以进行图片上的转换，如图12所示：



图12：DCGAN生成可以转换的图片。最左边的最上边生成的图是其下面五张图的平均值。最右边的也是如此。在这两者之间，不断改变下面五张图片的状态，求得的平均值可以解释为最上面两张图片的人脸变化状态。

3.1.3 个人观点

DCGAN 生成具有语义图片的成功能否用在自然语言处理上值得思考：

1. 联想 3.1.2.4 的功能，我们可以将 DCGAN 应用在自然语言处理上面，但是可能会存在以下两个问题：

（1）：对于网络架构，作者采用的是卷积网络，而卷积网络特别适用于图像的计算，将高维的像素点映射到低维像素点上在进行计算。但是对于一段文字来说，卷积将这一段文字的高维向量映射到低维空间上似乎没有什么意义。然而，神经网络可以模拟任何复杂的函数，我们也难以用理论证明它在网络里面到底是怎么完成任务的。拿最简单的图片分类问题举例，输入是像素矩阵，输出却是分类类别。所以，利用DCGAN生成文字是值得思考的。

（2）：（1）中质疑卷积网络是否适合文字的计算，那么我们不妨将卷积网络改为适合文字的网络结构，下面是对网络结构的设想：

- 一：生成器：RNN结构，判别器：RNN结构。
- 二：生成器：RNN结构，判别器：卷积结构或其他。
- 三：生成器：卷积结构，判别器：RNN结构。
- 四：生成器：卷积结构，判别器：卷积结构。

2. 联想3.1.2.3 的功能（控制生成图片里面的物体）：如果我们利用卷积网络的结构可以生成合理的文字的话，那完全可以通过控制filter来控制文字的生成。同样地，如果可以利用RNN结构生成合理文字的话，那我们可以控制RNN的中间步骤来改变生成的文字，因为某个中间的步骤可能反应的是一段文字里面某个文字的情况。

3.2 W-GAN

GAN在基于梯度下降训练时存在梯度消失的问题，导致优化目标不连续。为了解决训练梯度消失问题，Arjovsky 等^[25]提出了Wasserstein GAN (W-GAN). W-GAN 用Earth-Mover 代替Jensen-Shannon 散度来度量真实样本和生成样本分布之间的距离，用一个批评函数 f 来对应GAN的判别器，而且损失函数 f 需要建立在Lipschitz 连续性假设上。

3.2.1 Earth-Mover (Wasserstein) 距离

EM 距离替代 JS 散度，从而很好地解决了损失函数梯度消失的问题，下面我们来探讨一下 EM 距离的内容。EM 距离公式如下：

$$W(P_r, P_g) = \inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y) \sim \gamma} [\|x - y\|] \quad (10)$$

$\Pi(P_r, P_g)$ 是 P_r, P_g 组合起来的所有可能的联合分布的集合，反过来说， $\Pi(P_r, P_g)$ 中每一个分布的边缘分布都是 P_r 和 P_g 。对于每一个可能的联合分布 γ 而言，可以从中采样 $(x, y) \sim \gamma$ 得到一个真实样本 x 和一个生成样本 y ，并算出这对样本的距离 $\|x - y\|$ ，所以可以计算该联合分布 γ 下样本对距离的期望值 $E_{(x,y) \sim \gamma} [\|x - y\|]$ 。在所有可能的联合分布中能够对这个期望值取到的下界 $\inf_{\gamma \in \Pi(P_r, P_g)} E_{(x,y) \sim \gamma} [\|x - y\|]$ ，就定义为 EM 距离。总之，EM 距离表示的是，找到一个联合分布，使变量 x 与 y 在联合分布下距离的期望最小。图 13 给出了 EM 距离与 JS 散度比较的一个简单地例子。从图中可以看到，EM 距离随着 θ 值变化一直都有梯度，而 JS 散度则是一个常数。

作者又通过理论证明了 EM 距离可以由以下公式替换：

$$W(P_r, P_\theta) = \sup_{\|f\|_L \leq 1} E_{x \sim P_r} [f(x)] - E_{x \sim P_\theta} [f(x)] \quad (11)$$

而后作者又通过 Lipschitz 函数 f 的性质将公式 11 转换为公式 12

$$W(P_r, P_g) \approx \max_{w \in W} E_{x \sim p_r} [f_w(x)] - E_{z \sim p_z(z)} [f_w(g_\theta(z))] \quad (12)$$

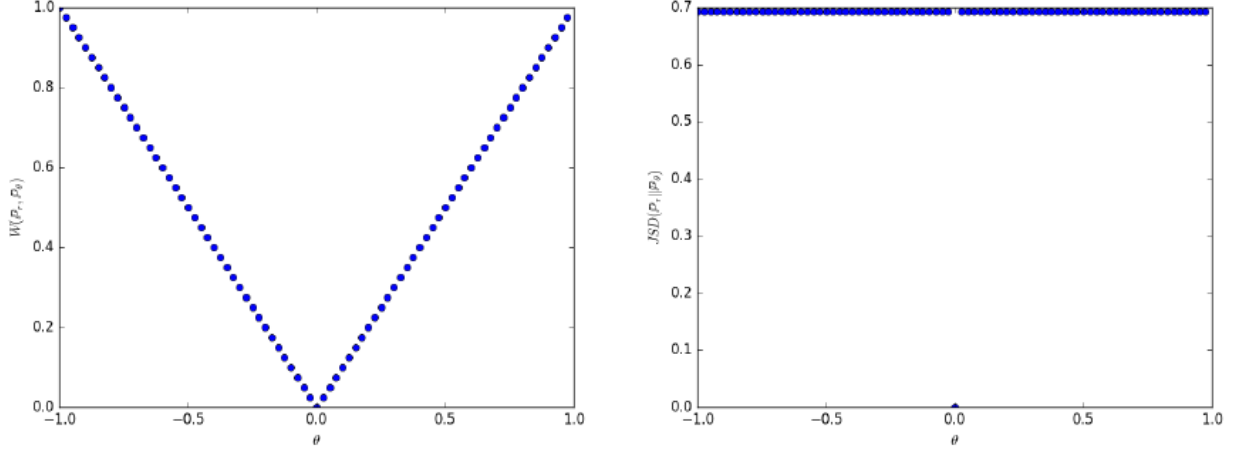


图 13. $Z \sim U[0, 1]$ 表示均匀分布。 P_0 表示 x 值为 0, y 值是任意值的变量, 而 P_θ 则表示 x 值为 θ , y 值是任意值得变量。从图中可以看到, 无论 θ 值如何变化, EM 具有梯度, 而 JS 散度为常数。

3.2.2 WGAN 生成器与判别器的损失函数及其训练算法

既然 EM 距离可以表示两个数据分布之间的差异, 它相对于 JS 散度又不会出现梯度消失的问题, 所以作者将 EM 距离改为生成器与判别器的损失函数 (公式 12)。判别器与生成器的损失函数分别如下:

$$\max E_{x \sim p_r}[f_w(x)] - E_{z \sim p_z(z)}[f_w(g_\theta(z))] \quad (13)$$

$$\text{Min } -E_{z \sim p_z(z)}[f_w(g_\theta(z))] \quad (14)$$

明确了损失函数之后, 我们就可以制定算法了。如下

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α , the learning rate. c , the clipping parameter. m , the batch size. n_{critic} , the number of iterations of the critic per generator iteration.

Require: : w_0 , initial critic parameters. θ_0 , initial generator's parameters.

```

1: while  $\theta$  has not converged do
2:   for  $t = 0, \dots, n_{\text{critic}}$  do
3:     Sample  $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$  a batch from the real data.
4:     Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples.
5:      $g_w \leftarrow \nabla_w [\frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))]$ 
6:      $w \leftarrow w + \alpha \cdot \text{RMSPProp}(w, g_w)$ 
7:      $w \leftarrow \text{clip}(w, -c, c)$ 
8:   end for
9:   Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples.
10:   $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))$ 
11:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSPProp}(\theta, g_\theta)$ 
12: end while

```

3.2.3 WGAN算法的局限性

要想用神经网络估计训练损失函数，作者发现神经网络的权值 w 需要固定在特别的范围内，即 W 每次更新完之后，范围控制在 $[-0.01, 0.01]$ 。权值修剪明显是一种糟糕的方法用来计算损失函数。如果系数很大，这就需要很长的时间修剪权值，从而降低了训练的效率，作者将这个主题遗留了下来，鼓励感兴趣的研究者改善这一问题。

3.2.4 WGAN的实验应用

3.2.4.1 Wasserstein距离的优势

作者通过比较生成图片质量来说明Wasserstein距离的重要性。实验用到的数据是LSUN卧室数据集，用到的模型是GAN, DCGAN以及应用卷积结构的GAN。生成的图片像素为 64×64 ，频道为3。试验结果如图14。

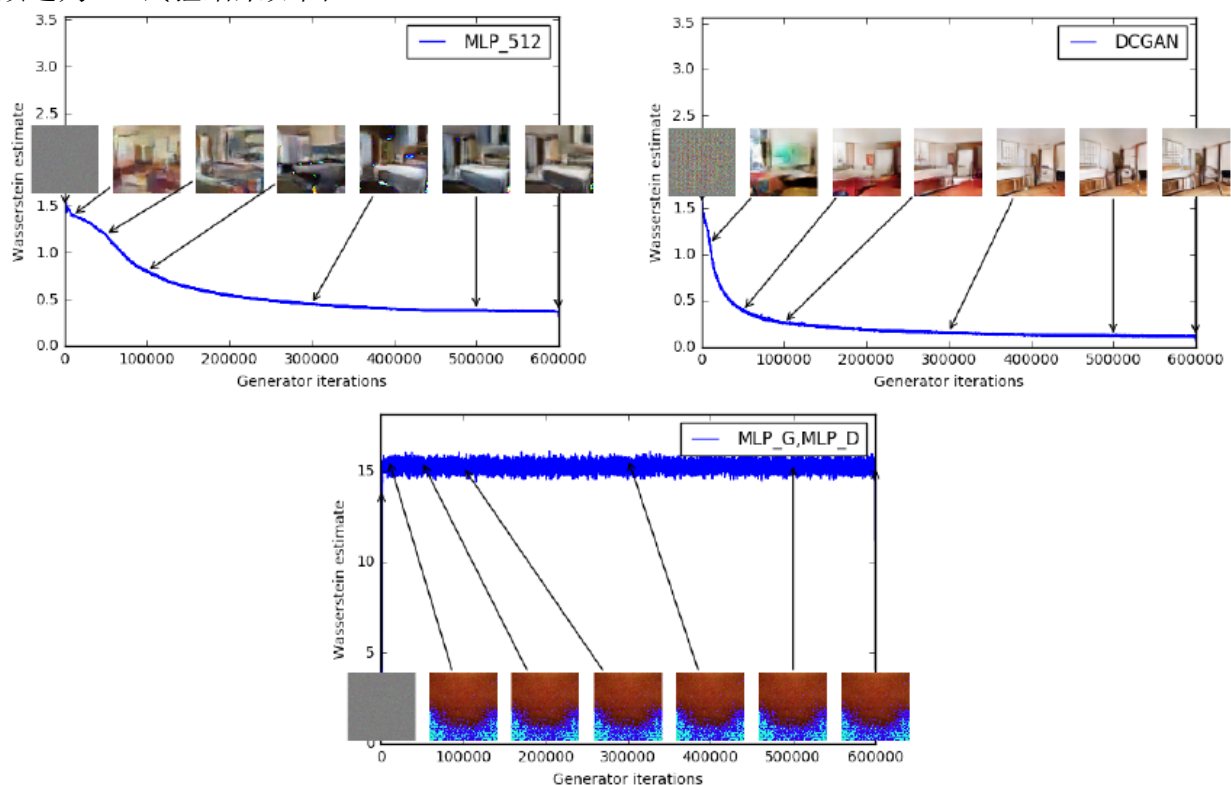


图14. 左上：生成器是MLP，4个隐藏层，每个层包含512个神经元。右上：生成器是标准的DCGAN。下边：生成器与判别器都是MLP，但是学习率（learning rate）很高，导致损失是常数。

3.2.4.2 WGAN生成图片的优势

作者通过多方面的比较来证明WGAN损失函数的优势，作者首先使用WGAN结合DCGAN的架构，拿生成的图片同DCGAN做做比较，效果相似，如图15



图15：左图是WGAN利用DCGAN结构。右侧是标准的DCGAN（GAN损失函数和卷积结构）

作者移除WGAN和DCGAN里面的批次标准化技术，DCGAN的效果就变的很差了，如图16

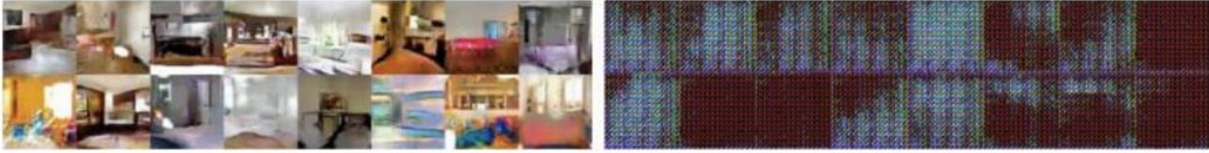


图16：左图是WGAN利用DCGAN结构。右侧是标准的DCGAN（GAN损失函数和卷积结构）

如果WGAN和原始GAN都使用多层全连接网络（MLP），相比较卷积结构，WGAN质量会变差些，但是原始GAN质量变得更差，如图17



图17：左图是WGAN利用MLP结构。右侧是GAN利用MLP结构

3.3 WGAN-GP

WGAN的原作者指明了其局限性，即权值设置在 $[-0.01, 0.01]$ 是非常糟糕的方法来模拟损失函数。这就意味了网络的权重在很小的范围内，对于深度神经网络来说不能充分发挥深度神经网络的拟合能力。并且，强制剪切权重容易导致梯度消失或者梯度爆炸。为了解决这个问题，并且找一个合适的方式满足lipschitz连续性条件，作者Gulrajani^[26]等人提出了使用梯度惩罚的方式以满足此连续性条件。

3.3.1 WGAN-GP生成器与判别器的损失函数及其训练算法

损失函数和效果分别如公式15、图18、以及图19：

$$\min_G \max_D E_{x \sim p_g}[D(x)] - E_{x \sim p_r}[D(x)] + E_{x \sim p_{x'}}[(\|\nabla_{x'} D(x')\|_2 - 1)^2] \quad (15)$$

$E_{x \sim p_g}[D(x)] - E_{x \sim p_r}[D(x)]$ 是WGAN的损失函数，作者在此基础上加了后半部分，即梯度惩罚(gradient penalty)， x' 是随机样本 $x' \sim p_{x'}$

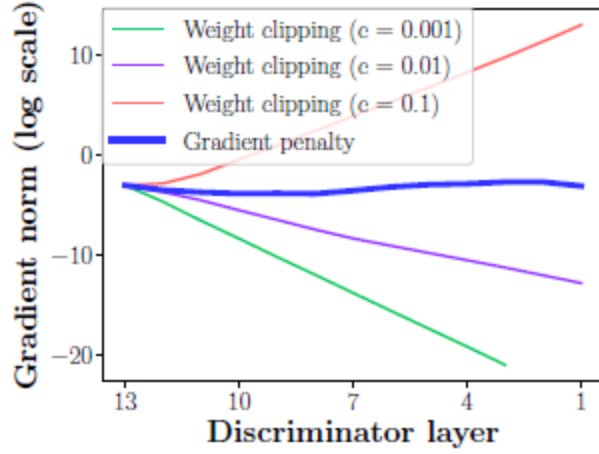


图18. 绿、紫、红线表示深度WGAN经过不同的权值修剪后，应用在数据集Swiss Roll的梯度范数。WGAN经过权值修剪后，要不发生梯度消失，要不发生梯度爆炸。但是WGAN-GP却不一样。

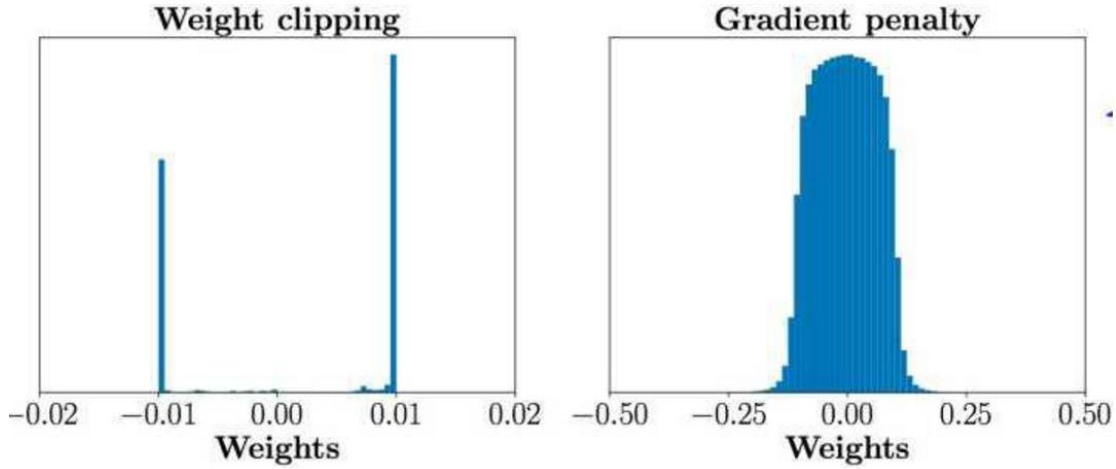


图19. 在发生权值修剪的过程中，WGAN大多数的权重都是两个极端情况，即-0.01和0.01。而WGAN-GP的权值范围很广。

从公式15可以看处，判别器的损失函数为公式(16)：

$$\max_D E_{x \sim p_g}[D(x)] - E_{x \sim p_r}[D(x)] + E_{x \sim p_{x'}}[(\|\nabla_{x'} D(x')\|_2 - 1)^2] \quad (16)$$

生成器的损失函数为

$$\min_G -E_{x \sim p_r}[D(x)] \quad (17)$$

明确了损失函数之后，我们就可以制定算法了。如下图20

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ , the number of critic iterations per generator iteration n_{critic} , the batch size m , Adam hyperparameters α, β_1, β_2 .

Require: initial critic parameters w_0 , initial generator parameters θ_0 .

```

1: while  $\theta$  has not converged do
2:   for  $t = 1, \dots, n_{\text{critic}}$  do
3:     for  $i = 1, \dots, m$  do
4:       Sample real data  $x \sim \mathbb{P}_r$ , latent variable  $z \sim p(z)$ , a random number  $\epsilon \sim U[0, 1]$ .
5:        $\tilde{x} \leftarrow G_\theta(z)$ 
6:        $\hat{x} \leftarrow \epsilon x + (1 - \epsilon)\tilde{x}$ 
7:        $L^{(i)} \leftarrow D_w(\tilde{x}) - D_w(x) + \lambda(\|\nabla_{\hat{x}} D_w(\hat{x})\|_2 - 1)^2$ 
8:     end for
9:      $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$ 
10:   end for
11:   Sample a batch of latent variables  $\{z^{(i)}\}_{i=1}^m \sim p(z)$ .
12:    $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^m -D_w(G_\theta(z)), \theta, \alpha, \beta_1, \beta_2)$ 
13: end while

```

图20.WGAN-GP算法

3.3.2 WGAN-GP的贡献

WGAN-GP提出了一种新的lipschitz连续性限制手法—梯度惩罚，用来解决WGAN的遗留问题。总之，WGAN-GP的贡献有以下几点：

1. 新的lipschitz连续性限制手法—梯度惩罚，解决了训练梯度消失梯度爆炸的问题。
2. WGAN-GP比标准WGAN拥有更快的收敛速度，并能生成更高质量的样本。
3. 提供稳定的GAN训练方式，几乎不需要怎么调参，成功训练多种针对图片生成和语言模型的GAN架构。

但是，由于每个batch中的每一个样本都做了梯度惩罚，因此判别器中不能使用批次标准化的方法(batch normalization)，但是可以使用其他标准化的方法，比如层标准化(Layer Normalization)、权值标准化(Weight Normalization)和实例标准化(Instance Normalization)，作者则采用了实例标准化和权值标准化。

3.3.3 WGAN-GP的实验应用

作者将不同的损失函数和不同的网络结构应用在LSUN卧室数据集上，比较它们生成图片的结果，效果如图21所示。





























DCGAN	LSGAN	WGAN (clipping)	WGAN-GP (ours)
Baseline (G : DCGAN, D : DCGAN)			
			
G : No BN and a constant number of filters, D : DCGAN			
			
G : 4-layer 512-dim ReLU MLP, D : DCGAN			
			
No normalization in either G or D			
			
Gated multiplicative nonlinearities everywhere in G and D			
			
tanh nonlinearities everywhere in G and D			
			
101-layer ResNet G and D			
			

图21. 不同的GAN结构用不同的损失函数进行训练。可以看到，只有WGAN-GP在每个网络结构中获得成功。

3.4 C-GAN

原始 GAN 提出，与其他生成式模型相比，GAN 这种竞争的方式不再要求一个假设的数据分布，而是使用一种分布直接进行采样，从而真正达到理论上可以完全逼近真实数据，这也是 GAN 最大的优势。然而，这种不需要预先建模的方法缺点是太过自由了，对于较大的图片，较多的 pixel 的情形，基于简单 GAN 的方式就不太可控了。为了解决 GAN 太过自由这个问题，一个很自然的想法是给 GAN 加一些约束，于是Mirza^[27]等人提出了CGAN。这项工作提出了一种带条件约束的 GAN，在生成模型和判别模型的建模中均引入条件变量 y ，使用额外信息 y 对模型增加条件，可以指导数据生成过程。这些条件变量 y 可以基于多种信息，例如类别标签，用于图像修复的部分数据，来自不同模态的数据。如果条件变量 y 是类别标签，可以看做 CGAN 是把纯无监督的 GAN 变成有监督的模型的一种改进。这个简单直接的改进被证明非常有效，并广泛用于后续的相关工作中^[28]。

3.4.1 C-GAN 生成器与判别器的损失函数及其训练算法

条件生成式对抗网络（CGAN）是对原始 GAN 的一个扩展，生成器和判别器都增加额外信息 y 为条件， y 可以使任意信息，例如类别信息，或者其他模态的数据。如 Figure 1 所示，通过将额外信息 y 输送给判别模型和生成模型，作为输入层的一部分，从而实现条件 GAN。在生成模型中，先验输入噪声 $p(z)$ 和条件信息 y 联合组成了联合隐层表征。类似地，条件 GAN 的目标函数是带有条件概率的极小极大值博弈。

$$\min_G \max_D E_{x \sim p_{data}(x)} [\log D(x|y)] + E_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (18)$$

判别器的损失函数为：

$$\max_D E_{x \sim p_{data}(x)} [\log D(x|y)] + E_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (19)$$

对应地，生成器的损失函数为：

$$\min_G E_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (20)$$

可以看到，CGAN 的损失函数与原始 GAN 的极为相似，所以算法也极为的相似。

3.4.2 C-GAN 的网络结构

同 GAN 相比，CGAN 的输入多了条件变量 y ，生成器和判别器的网络结构如下图所示：

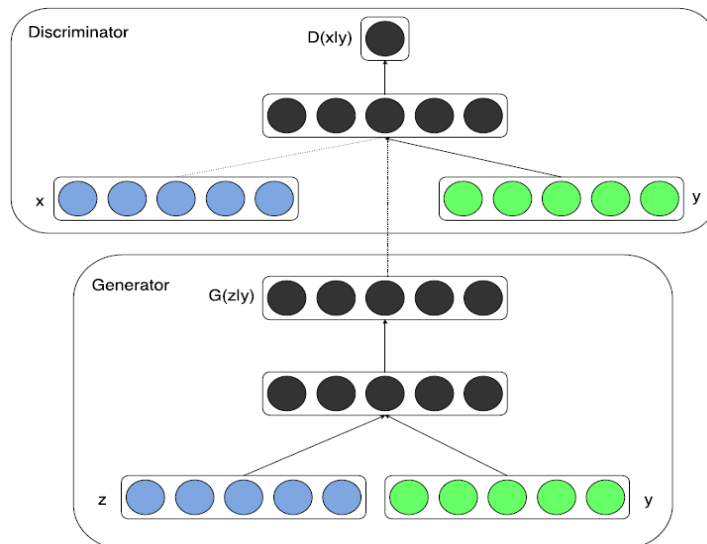


图22.C-GAN判别器生成器网络结构

3.4.3 C-GAN 的实验应用

3.4.3.1 CGAN 在 MNIST 数据集上的应用

在 MNIST 上以类别标签为条件（one-hot 编码）训练条件 GAN，可以根据标签条件信息，生成对应的数字。生成模型的输入是 100 维服从均匀分布的噪声向量，条件变量 y 是类别标签的编码。噪声 z 和标签 y 分别映射到隐层（200 和 1000 个单元），在映射到第二层前，联合所有单元。最终有一个 sigmoid 生成模型的输出（784 维），即 28×28 的单通道图像。

判别模型的输入是 784 维的图像数据和条件变量 y （类别标签的 one-hot 编码），输出是该样本来自训练集的概率。如下图所示：



图 23. 每一行是在 1 个 label 下的条件概率。每一列则是生成的图片

3.4.3.2 CGAN 图像自动标注（这个吊）

作者通过 Flickr 数据集进行试验。Flickr 图像数据集具有用户标签，表示的是图像的实际意义。作者首先训练一个卷积模型和 skip-gram 模型用来提取图像的特征和标签特征。对于生成器，输入的是隐变量和图像的特征 (y)。对于判别器，输入的是标签特征 (x) 和图像的特征。训练好后，生成器就可以对指定的图片生成它的标签了。效果图如图 24。

3.4.4 未来的工作

作者指出，实验结果是非常初级的，但是这些结果的确展示了 CGAN 强大的研究价值和应用。作者希望呈现更复杂的模性和更全面的分析来提高实验的表现。举例来说，在图像自动标注的实验中，作者只采用了单独的用户标签，但是如果同时采取多种用户标签可能会实现更好的结果。

另外，未来的工作会建立一个联合的训练计划去学校语言模型。





	User tags + annotations	Generated tags
	montanha, trem, inverno, frio, people, male, plant life, tree, structures, transport, car	taxi, passenger, line, transportation, railway station, passengers, railways, signals, rail, rails
	food, raspberry, delicious, homemade	chicken, fattening, cooked, peanut, cream, cookie, house made, bread, biscuit, bakes
	water, river	creek, lake, along, near, river, rocky, treeline, valley, woods, waters
	people, portrait, female, baby, indoor	love, people, posing, girl, young, strangers, pretty, women, happy, life

图24：产生的标签例子。对于多个用户标签，作者将它们分开并使它们——对应于图片（即每张图片只有一个用户标签）。作者将训练好的生成器生成100份标签，选择20个，这20个用户标签距离最近的，然后在20个里面选择10个最普通的单词作为最终的标签。

3.4.5 个人观点

1. C-GAN的实质是学习的是条件分布 $x \sim P(x|y)$ ，就像上面强大的图像自动标注应用一样，生成器学的是在特定图片特征下的用来表示图像的标签分布。如果这个延伸到时间序列上可能会有很好的研究价值($x \sim P(x|T)$)，如何用有意义的向量来表示时间是其中的关键。
2. C-GAN能够直接或者间接地控制生成的物体，这一点创新很值得研究。

3.5 LAPGAN(拉普拉斯金字塔生成式对抗网络)

前面讲到的 CGAN 使用额外信息 y 对模型增加条件，限制 GAN 生成数据的随意性，指导

数据生成过程。而 LAPGAN 是从另一个角度限制 GAN 生成数据的随意性，作者 Denton 等人^[29]的目的是不要让 GAN 一次完成全部任务，而是一次生成一部分，分多次生成一张完整的图片。

3.5.1 拉普拉斯金字塔

拉普拉斯金字塔是一个线性可逆的图像表示，该表示包含一套图像。假设我们建立一个高斯金字塔 $G(I) = [I_0, I_1, \dots, I_K]$ 。K 代表金字塔的级别，也就是说，越往上走，图像的像素越来越小。为了让 I_k, I_{k+1} 图像的规模一致，简单地做法是对 I_{k+1} 上采样。我们定义：

$$\mathbf{h}_k = \mathbf{G}_k(\mathbf{I}) - u(\mathbf{G}_{k+1}(\mathbf{I})) = \mathbf{I}_k - u(\mathbf{I}_{k+1}) \quad (21)$$

\mathbf{h}_k 表示的是 \mathbf{I}_{k+1} 图像经过上采样与 \mathbf{I}_k 图像的差值

$$\mathbf{I}_k = u(\mathbf{I}_{k+1}) + \mathbf{h}_k \quad (22)$$

3.5.2 LAPGAN 生成图片的过程

首先设置 $\mathbf{I}_{k+1}=0$ ，然后利用生成模型在级别 \mathbf{G}_k 上用隐变量 \mathbf{z}_k 产生残差图像 \mathbf{I}_k ： $\mathbf{I}_k = \mathbf{G}_k(\mathbf{z}_k)$ 。在这里需要注意的是所有的生成模型除了最高级别 \mathbf{G}_k 都是条件生成模型，他们都将上采样的图像作为输入的一部分，而另一部分的输入则是隐变量 \mathbf{z}_k 。 \mathbf{G}_k 的输出则是图像 \mathbf{I}_k 与 \mathbf{I}_{k+1} 上采样的差值 (见公式 23)。图 25 详细展示了生成器生成像素为 64X64 图像的过程 (K=3, 拥有 4 个生成模型)。

$$\tilde{\mathbf{I}}_k = u(\tilde{\mathbf{I}}_{k+1}) + \tilde{\mathbf{h}}_k = u(\tilde{\mathbf{I}}_{k+1}) + \mathbf{G}_k(\mathbf{z}_k, u(\tilde{\mathbf{I}}_{k+1})) \quad (23)$$

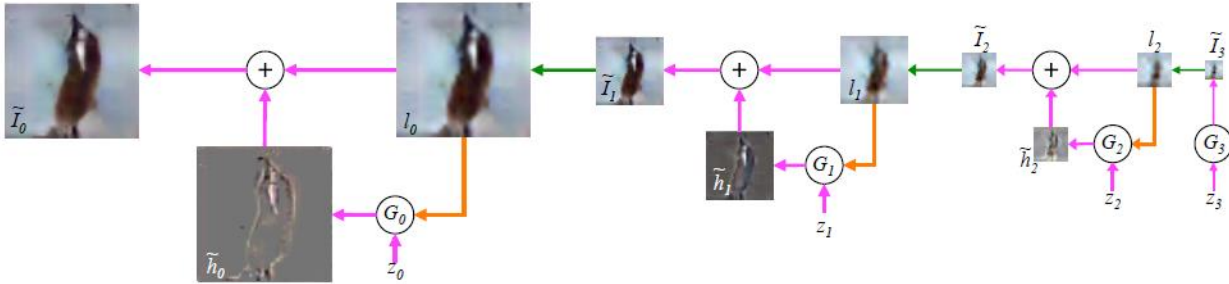


图 25. LAPGAN 生成图片的过程

3.5.3 LAPGAN 的博弈过程（训练过程）

这是多个生成器和多个判别器的博弈过程。如图 27 所示。每一个生成器和判别器的结构都是卷积的结构。注意 $\mathbf{L}_k = u(\mathbf{I}_{k+1})$ 。 \mathbf{D}_k 输入的是 \mathbf{h}_k (真实的差值)、 $\tilde{\mathbf{h}}_k$ (生成的差值) 以及 \mathbf{L}_k 并根据哲学条件信息来判断图像是否真实。到最后的一个步骤，图像足够小，就不像前面的步骤那样，判别器只有两个输入。

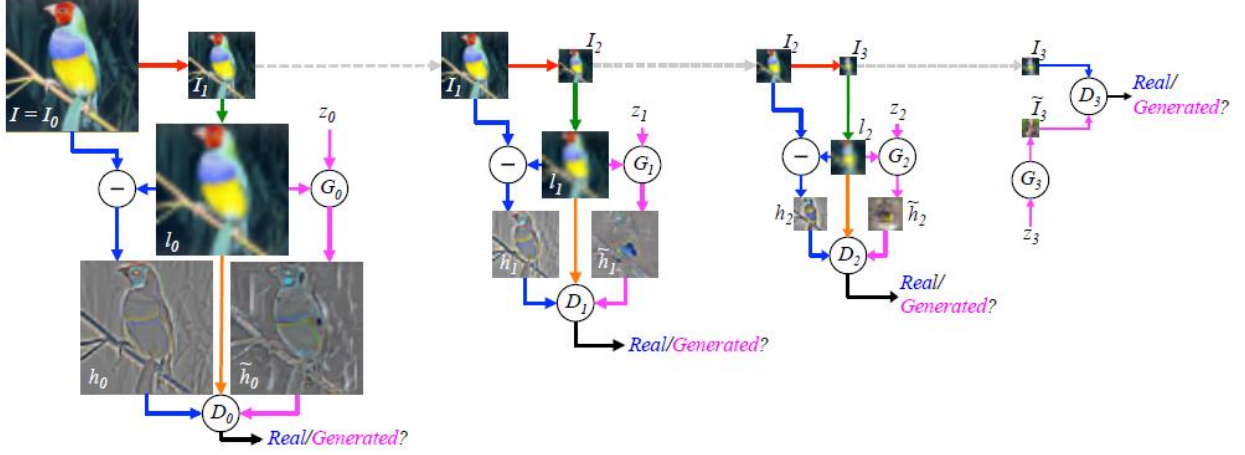


图 26. 生成器与判别器的博弈过程(训练过程)

3.5.4 LAPGAN 生成器与判别器的损失函数及其训练算法

作者没有给损失函数，但作者指出 LAPGAN 与 CGAN 的损失函数相似，由于 LAPGAN 每一步包括一个生成器与判别器，博弈过程一模一样(除最后一步，最后一步的损失函数应该与 GAN 相似)，结合上图我们可以推断出其损失函数如下：

$$\min_G \max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x|y)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (24)$$

在这里，y 是指 L_k (如图 27)。判别器和生成器的损失函数分别如下：

$$\max_D \mathbb{E}_{x \sim p_{data}(x)} [\log D(x|y)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (25)$$

$$\min_G \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z|y)))] \quad (26)$$

算法与 CGAN 的一模一样，不过最后一步应该与 GAN 的一样。

3.5.5 LAPGAN 的实验应用

作者使用类别条件的 LAPGAN, LAPGAN, GAN 在 CIFAR10 数据集上进行实验，效果图如图 27，图 28。

3.5.6 个人观点

作者指出，该模型的特点是每个级别的独立训练过程防止模型记忆训练数据集。但是训练过程复杂，而且作者只比较了 LAPGAN 与 GAN 的实验效果。既然 LAPGAN 在 CGAN 的基础上建立的，为什么不比较 CGAN 的效果。

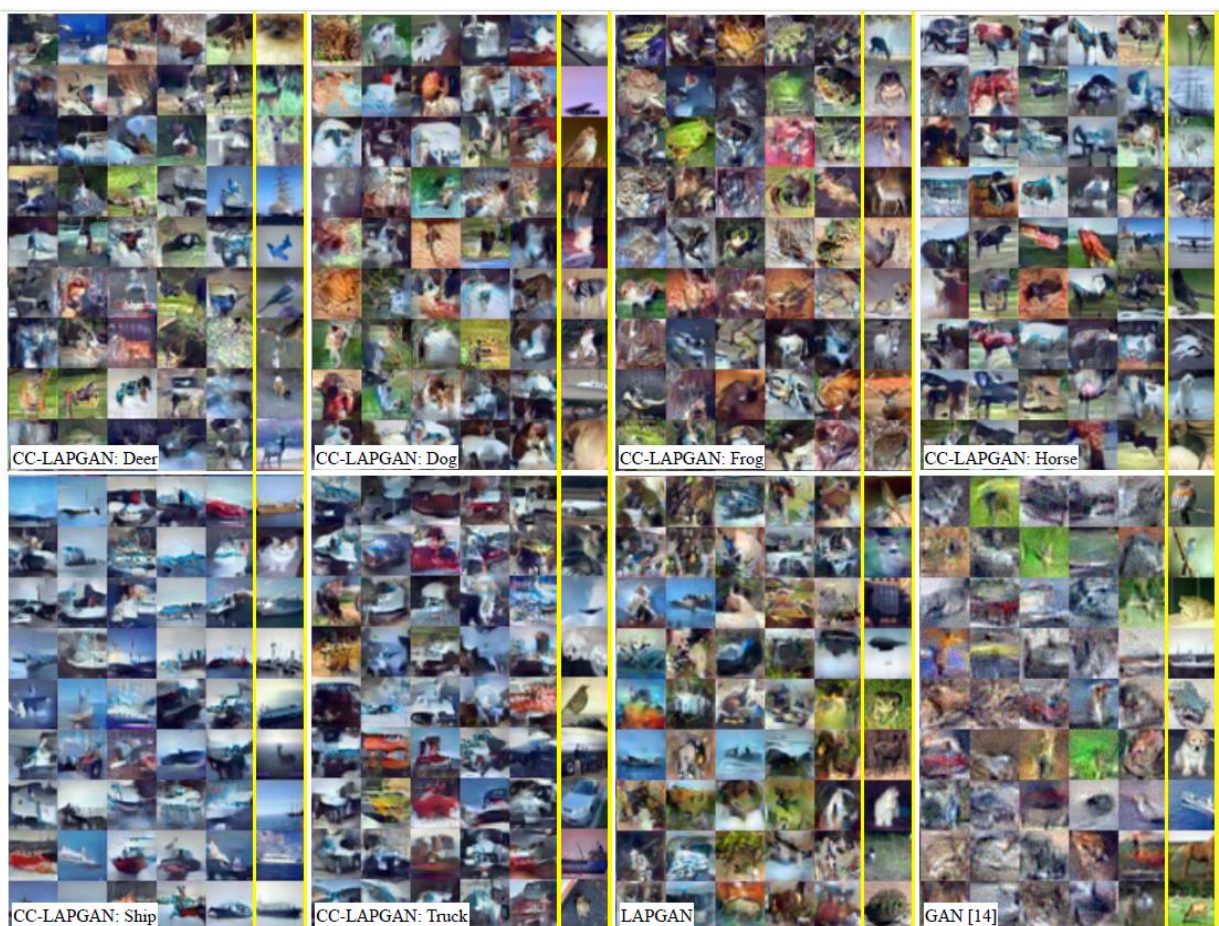


图 27. CIFAR10 例子：CC-LAPGAN, LAPGAN, 以及 GAN 模型比较

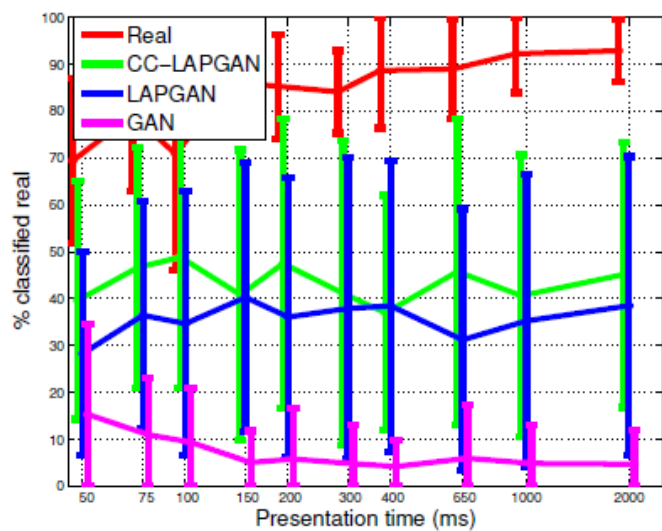


图 28. 横轴表示人类思考时间，纵轴表示判断的准确率

3.6 Info-GAN

InfoGAN的目的是试图解释隐变量。前面提到GAN的隐变量服从某种分布，但是这个分布背后的含义却不得而知。虽然经过训练的GAN可以生成新的图像，但是它却无法解决一个问题——生成具有某种特征的图像。例如，对于MNIST的数据，如果想要生成某个具体数字的图像，或者生成笔画较粗、方向倾斜的图像等，这时就会发现GAN无法解决这样的问题。Xi Chen^[30] 等人为了解决这样的问题，从信息论的角度，尝试解决GAN隐变量可解释性问题。

3.6.1 互信息

从字面意思可以判断出它衡量了随机变量之间的关联关系。假设我们已知 x ，需要知道 y 在 x 条件下的不确定性，这就要用到互信息的信息。互信息的公式如下：

$$I(X;Y) = \int_{x \in X} \int_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} dx dy, \quad (27)$$

经过如下变换,可得到:

$$\begin{aligned} I(X;Y) &= \int_{x \in X} \int_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} dx dy \\ &= \int_{x \in X} \int_{y \in Y} P(x,y) \log \frac{P(x,y)}{P(x)} dx dy - \int_{x \in X} \int_{y \in Y} P(x,y) \log P(y) dx dy \\ &= \int_{x \in X} \int_{y \in Y} P(x)P(y|x) \log P(y|x) dx dy - \int_{y \in Y} \log P(y) \int_{x \in X} P(x,y) dx dy \\ &= \int_{x \in X} P(x) \int_{y \in Y} P(y|x) \log P(y|x) dx dy - \int_{y \in Y} \log P(y) P(y) dx dy \\ &= - \int_{x \in X} P(x) H(Y|x) dx + H(Y) \\ &= H(Y) - H(Y|X) \end{aligned}$$

$$I(X;Y) = H(Y) - H(Y|X) = H(X) - H(X|Y) \quad (28)$$

用通俗的话讲， $I(X;Y)$ 表示的是 X 与 Y 之间的关系，它的值等于变量 Y 的不确定性与在 x 已知的情况下 Y 的不确定性的差值。也就是说，它表示在 X 已知的情况下， Y 不确定性减少了多少。类似的，它的值等于变量 Y 的不确定性与在 x 已知的情况下 Y 的不确定性的差值。

3.6.2 InfoGAN 生成器与判别器的损失函数及其训练算法

如果 X 与 Y 相互独立， $I(X;Y)=0$ ，因为两者毫无关系。相比较而言，如果 X 与 Y 关系越紧密，则 $I(X;Y)$ 的值越大。这种表示关系形成了一种新的损失函数：给定任意的 $x \sim P_G(x)$ ，我们想让 $P_G(c|x)$ 的交叉熵最小(即 $I(X;Y)$ 的值最大)。换句话说，隐变量 c 的信息在生成过程中不应该丢失。在这里，隐变量分为两部分(z, c)，一部分隐变量 z 表示与生成的物体无关，而另一部分变量 c 则表示与生成的物体相关。所以生成器不想让与生成的物体相关的隐变量 c 丢失信息。整体的目标函数如下：

$$\min_G \max_D V_I(D, G) = V(D, G) - \lambda I(c; G(z, c)) \quad (29)$$

其中， $V(D, G)$ 表示GAN的损失函数。

3.6.2.1. $I(c; G(z, c))$ 的计算

从公式22已知， $I(c; G(z, c)) = I(c; x) = H(c) - H(c|x)$ ，求解公式如下：

$$\begin{aligned} I(c; G(z, c)) &= H(c) - H(c|G(z, c)) \\ &= \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log P(c'|x)]] + H(c) \\ &= \mathbb{E}_{x \sim G(z, c)} [\underbrace{D_{\text{KL}}(P(\cdot|x) \parallel Q(\cdot|x))}_{\geq 0} + \mathbb{E}_{c' \sim P(c|x)} [\log Q(c'|x)]] + H(c) \\ &\geq \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log Q(c'|x)]] + H(c) \end{aligned}$$

这种求互信息下界的方法就是变化信息最大化 (Variational Information Maximization)

引理：对于任意的变量 X, Y 并且函数 $f(x, y)$ 在合适的正则条件下：

$$\mathbb{E}_{x \sim X, y \sim Y} [f(x, y)] = \mathbb{E}_{x \sim X, y \sim Y | x, x' \sim X | y} [f(x', y)].$$

通过使用引理，可以定义互信息的下界为：

$$\begin{aligned} L_I(G, Q) &= \mathbb{E}_{c \sim P(c), x \sim G(z, c)} [\log Q(c|x)] + H(c) \\ &= \mathbb{E}_{x \sim G(z, c)} [\mathbb{E}_{c' \sim P(c|x)} [\log Q(c'|x)]] + H(c) \\ &\leq I(c; G(z, c)) \end{aligned} \quad (30)$$

因此，损失函数可以变换为：

$$\min_{G, Q} \max_D V_{\text{InfoGAN}}(D, G, Q) = V(D, G) - \lambda L_I(G, Q) \quad (31)$$

因为Info的损失函数包含了GAN的损失函数，所以两者算法类似。

3.6.3 InfoGAN的实验应用

模型在训练前定义了12个和图像有强烈互信息的随机变量，其中10个变量表示显示的数字，它们组成一个Categorical的离散随机向量；另外2个是服从范围为 $[-1, 1]$ 的连续随机变量。训练完成后，调整离散随机变量输入并生成图像，得到如图25所示的数字图像。

可以看出模型很好地识别了这些数字。调整另外两个连续随机变量，可以生成如图26所示的数字图像。可以看出，这两个连续随机变量学到了数字粗细和倾斜的特征，而且这是在完全没有暗示的情况下完成的。

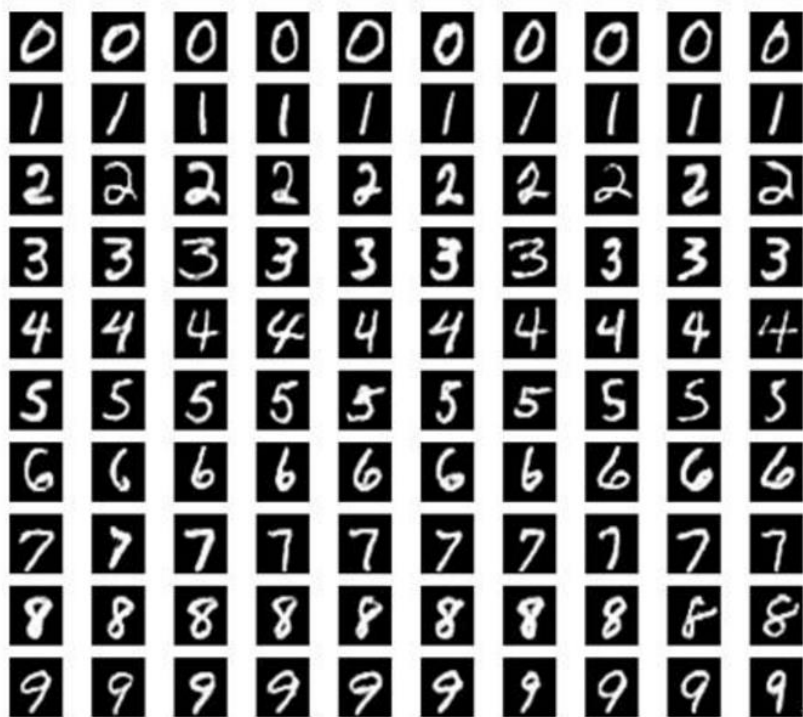


图29. 10个离散随机变量对生成数字的影响

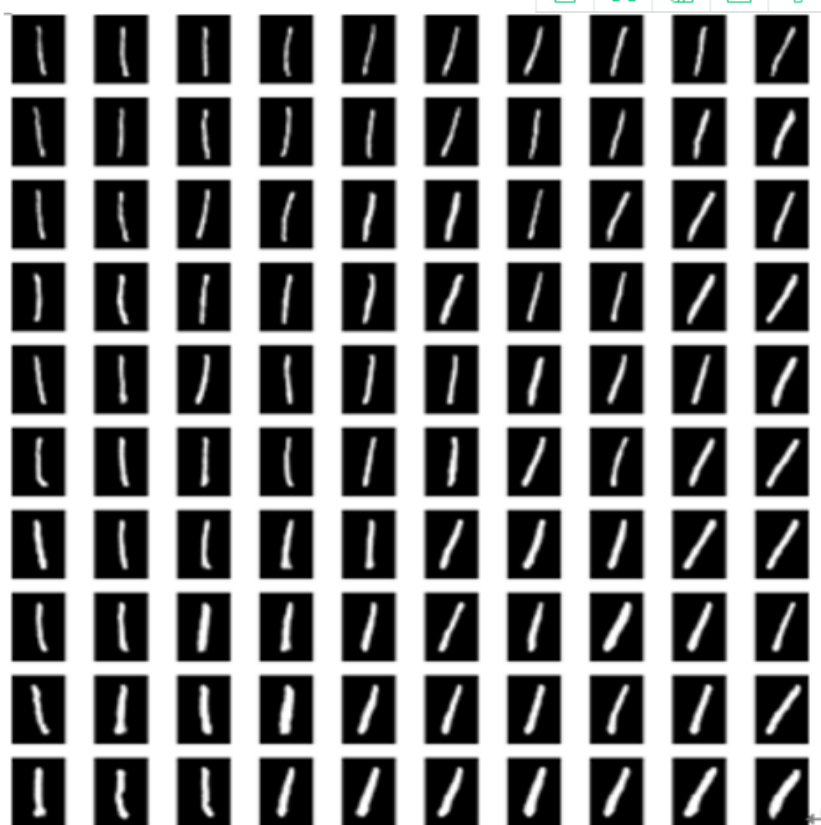


图30. 2个连续随机变量对生成数字的影响

3.7. SeqGAN

下面我们来看一下利用GAN思想和强化学习的思想来生成文字的模型-SeqGAN。Lantao Yu^[31]将政策(policy)网络当做生成器，如图31所示，已经存在的红色圆点称为现在的状态(state)，要生成的下一个红色圆点称作动作(action)，因为D需要对一个完整的序列评分，所以就是用MCTS(蒙特卡洛树搜索)将每一个动作的各种可能性补全，D对这些完整的序列产生reward，回传给G，通过增强学习更新G。这样就是用强化学的方式，训练出一个可以产生下一个最优的action的生成网络。

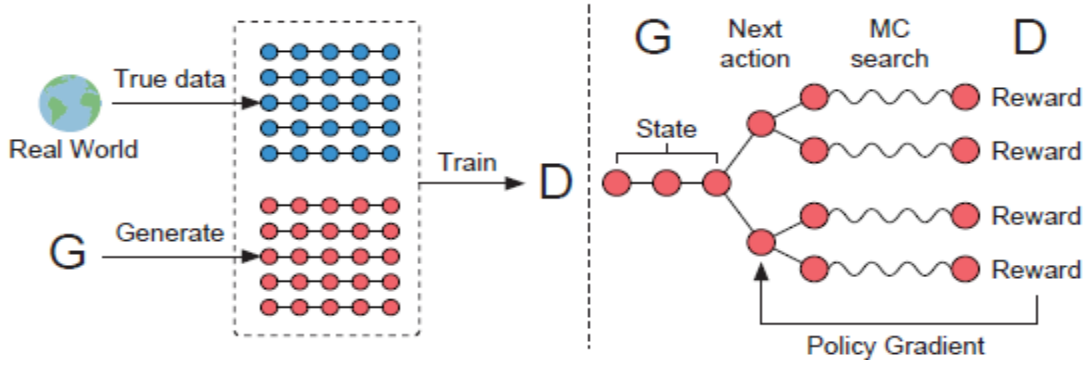


图31. SeqGAN网络结构图

3.7.1 SeqGAN生成器与判别器的损失函数及其训练算法

生成器G的目标是生成文字序列具有最大化奖励的期望。其目标函数如下：

$$J(\theta) = \mathbb{E}[R_T | s_0, \theta] = \sum_{y_1 \in \mathcal{Y}} G_\theta(y_1 | s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0, y_1) \quad (32)$$

在这里 $J(\theta)$ 为奖励的期望， R_T 为整个句子的期望。期望来自于判别器 D_ϕ ， $Q_{D_\phi}^{G_\theta}(s, a)$ 是一个文字序列的action-value函数。它代表的是文字序列从初始，采取行动 a ，遵从政策 G_θ 的累积奖励的期望。而生成器的目的是产生序列让判别器认为它是真实的。

下面的问题就是函数action-value怎么求，作者利用了强化学习的方法来计算并把这个Q值看作是判别D的返回值

$$Q_{D_\phi}^{G_\theta}(a = y_T, s = Y_{1:T-1}) = D_\phi(Y_{1:T}). \quad (33)$$

$Y_{1:T} = (y_1, \dots, y_T)$ 时间1到T的整个文字序列。而作者使用蒙特卡洛搜索找到所有的序列的可能，然后计算期望奖励。如下图所示：

$$Q_{D_\phi}^{G_\theta}(s = Y_{1:t-1}, a = y_t) = \frac{1}{N} \sum_{n=1}^N D_\phi(Y_{1:T}^n), \quad Y_{1:T}^n \in \text{MC}^{G_\theta}(Y_{1:t}; N) \quad (34)$$

在这里， G_θ 表示政策(Policy)， N 表示从头到尾按照政策生成所有的文字序列，这个过程进行 N 次。就这样，我们生成了一些逼真的文字序列。我们就要用如下方式训练D，让D尽可能地判断这段文字假，也就是让期望奖励最小：

$$\min_{\phi} -\mathbb{E}_{Y \sim p_{\text{data}}} [\log D_\phi(Y)] - \mathbb{E}_{Y \sim G_\theta} [\log(1 - D_\phi(Y))] \quad (35)$$

D训练了一轮或者多轮之后，就得到了一个更优秀的D，此时要用D去更新G

$$\nabla_{\theta} J(\theta) = \sum_{t=1}^T \mathbb{E}_{Y_{1:t-1} \sim G_{\theta}} \left[\sum_{y_t \in \mathcal{Y}} \nabla_{\theta} G_{\theta}(y_t | Y_{1:t-1}) \cdot Q_{D_{\phi}}^{G_{\theta}}(Y_{1:t-1}, y_t) \right] \quad (36)$$

而后作者建立了估值方法用来计算公式(36)

$$\begin{aligned} \nabla_{\theta} J(\theta) &\simeq \sum_{t=1}^T \sum_{y_t \in \mathcal{Y}} \nabla_{\theta} G_{\theta}(y_t | Y_{1:t-1}) \cdot Q_{D_{\phi}}^{G_{\theta}}(Y_{1:t-1}, y_t) \\ &= \sum_{t=1}^T \sum_{y_t \in \mathcal{Y}} G_{\theta}(y_t | Y_{1:t-1}) \nabla_{\theta} \log G_{\theta}(y_t | Y_{1:t-1}) \cdot Q_{D_{\phi}}^{G_{\theta}}(Y_{1:t-1}, y_t) \\ &= \sum_{t=1}^T \mathbb{E}_{y_t \sim G_{\theta}(y_t | Y_{1:t-1})} [\nabla_{\theta} \log G_{\theta}(y_t | Y_{1:t-1}) \cdot Q_{D_{\phi}}^{G_{\theta}}(Y_{1:t-1}, y_t)], \end{aligned} \quad (37)$$

所以我们可以按如下方式更新生成器的系数

$$\theta \leftarrow \theta + \alpha_h \nabla_{\theta} J(\theta), \quad (38)$$

ah表示在第h步的学习率。

知道了损失函数，下面我们看看是如何训练的，如图32

Algorithm 1 Sequence Generative Adversarial Nets

Require: generator policy G_{θ} ; roll-out policy G_{β} ; discriminator D_{ϕ} ; a sequence dataset $\mathcal{S} = \{X_{1:T}\}$

- 1: Initialize G_{θ} , D_{ϕ} with random weights θ, ϕ .
- 2: Pre-train G_{θ} using MLE on \mathcal{S}
- 3: $\beta \leftarrow \theta$
- 4: Generate negative samples using G_{θ} for training D_{ϕ}
- 5: Pre-train D_{ϕ} via minimizing the cross entropy
- 6: **repeat**
- 7: **for** g-steps **do**
- 8: Generate a sequence $Y_{1:T} = (y_1, \dots, y_T) \sim G_{\theta}$
- 9: **for** t in $1 : T$ **do**
- 10: Compute $Q(a = y_t; s = Y_{1:t-1})$ by Eq. (4)
- 11: **end for**
- 12: Update generator parameters via policy gradient Eq. (8)
- 13: **end for**
- 14: **for** d-steps **do**
- 15: Use current G_{θ} to generate negative examples and combine with given positive examples \mathcal{S}
- 16: Train discriminator D_{ϕ} for k epochs by Eq. (5)
- 17: **end for**
- 18: $\beta \leftarrow \theta$
- 19: **until** SeqGAN converges

图32. seqGAN训练损失函

算法图里的公式 4, 8, 5 分别表示文中公式 34, 38, 35。

首先随机初始化 G 网络和 D 网络参数。

通过 MLE 预训练 G 网络，目的是提高 G 网络的搜索效率。

使用预训练的 G 生成一些数据，用来通过最小化交叉熵来预训练 D。

1. 开始生成文字序列，并使用公式 34 计算奖励（这个奖励来自于 G 生成的文字序列与 D 产生的 Q 值）。
 2. 使用公式 38 更新 G 的参数。
 3. 更优的 G 生成更好的文字序列，和真实数据一起通过公式 35 训练 D。
- 以上 1, 2, 3 循环训练直到收敛。

3.7.2 SeqGAN 的实验应用

作者采用的语言的数据集是中国古诗集合奥巴马政治演讲集，实验结果如图 33, 34, 其中，BLEU 指标表现惊人，甚至超过了真实数据。

Algorithm	BLEU-3	<i>p</i> -value	BLEU-4	<i>p</i> -value
MLE	0.519	$< 10^{-6}$	0.416	0.00014
SeqGAN	0.556		0.427	

图 33. SeqGAN 同其他算法在中国古诗集的评判比较

Algorithm	Human score	<i>p</i> -value	BLEU-2	<i>p</i> -value
MLE	0.4165	0.0034	0.6670	$< 10^{-6}$
SeqGAN	0.5356		0.7389	
Real data	0.6011		0.746	

图 34. SeqGAN 同其他算法在奥巴马政治演讲集的评判比较

3.8 LSGAN、Semi-GAN、BiGANs、ACGAN

Qi^[32]提出了 Loss-sensitive GAN (LS-GAN)，LS-GAN 的提出改善了 GAN 的损失函数，它将最小化目标函数得到的损失函数限定在满足 Lipschitz 连续性函数类上，作者还给出了梯度消失时的定量分析结果。

Odena^[33]提出了 Semi-GAN，将真实数据的标注信息加入判别器 D 的训练。与 CGAN 的想法相似。

Donahue 等^[34]提出一种 Bidi-rectional GANs (BiGANs) 来实现将复杂数据映射到隐变量空间，从而实现特征学习。

Odena 等^[35]提出的 Auxiliary Classifier GAN (AC-GAN) 可以实现多分类问题，它的判别器输出相应的标签概率。在实际训练中，目标函数则包含真实数据来源的似然和正确分类标签的似然，不再单独由判别器二分类损失来反传调节参数，可以进一步调节损失函数使得分类正确率更高，AC-GAN 的关键是可以利用输入生成器的标注信息来生成对应的图像标签，同时还可以在判别器扩展调节损失函数，从而进一步提高对抗网络的生成和判别能力。

4. GAN 算法的比较

Google 大脑^[36] 发布了一篇论文，比较了各类型的 GAN 在四种数据集上的表现，效果图如下图：

	MNIST	FASHION	CIFAR	CELEBA
MM GAN	9.8 ± 0.9	29.6 ± 1.6	72.7 ± 3.6	65.6 ± 4.2
NS GAN	6.8 ± 0.5	26.5 ± 1.6	58.5 ± 1.9	55.0 ± 3.3
LSGAN	7.8 ± 0.6*	30.7 ± 2.2	87.1 ± 47.5	53.9 ± 2.8*
WGAN	6.7 ± 0.4	21.5 ± 1.6	55.2 ± 2.3	41.3 ± 2.0
WGAN GP	20.3 ± 5.0	24.5 ± 2.1	55.8 ± 0.9	30.0 ± 1.0
DRAGAN	7.6 ± 0.4	27.7 ± 1.2	69.8 ± 2.0	42.3 ± 3.0
BEGAN	13.1 ± 1.0	22.9 ± 0.9	71.4 ± 1.6	38.9 ± 0.9
VAE	23.8 ± 0.6	58.7 ± 1.2	155.7 ± 11.6	85.7 ± 3.8

图 35. 评判指标为 FID 分数。计算过程分为三步。第一步：通过广泛查找超参数的方法找到最优的超参数组合。第二步：训练模型 50 次，每一次训练具有最优的超参数和不同的初始化权值。第三步：对 50 次的 FID 进行平均化的计算和标准差计算。

其中，FID 表达的是一种距离关系：

$$\text{FID}(x, g) = \|\mu_x - \mu_g\|_2^2 + \text{Tr}(\Sigma_x + \Sigma_g - 2(\Sigma_x \Sigma_g)^{\frac{1}{2}}), \quad (39)$$

x 与 g 分别表示真实数据和生成的数据， (μ_x, Σ_x) 与 (μ_g, Σ_g) 分别表示真实数据和生成数据的平均值和协方差。*表示出现失败的运行，通常是模型的崩溃和训练的失败。

通过比较，作者给出了一些很重要的结论。(1) 每种 GAN 并没有严格意义上的优于其他的 GAN。(2) 越大型的网络结构会改善所有的 GAN。(3) 每个模型的表现严重依赖数据集的不同。

5. GAN 的应用领域

作为一个具有无限生成能力的模型，GAN 的直接应用就是建模，生成与真实数据分布一致的数据样本，例如可以生成图像、视频等。GAN 可以用于解决标注数据不足时的学习问题，例如无监督学习、半监督学习等。GAN 还可以用于语音和语言处理，例如生成对话、由文本生成图像等。本节从图像和视觉、语音和语言、其他领域三个方面来阐述 GAN 的应用。

5.1 图像和视觉领域

GAN 能够生成与真实数据分布一致的图像。一个典型应用来自 Twitter 公司，Ledig 等提出利用 GAN 来将一个低清模糊图像变换为具有丰富细节的高清图像。作者用 VGG 网络作为

判别器,用参数化的残差网络[1表示生成器,实验结果如图36所示,可以看到GAN生成了细节丰富的图像。GAN也开始用于生成自动驾驶场景。Santana等[提出利用GAN来生成与实际交通场景分布一致的图像,再训练一个基于RNN的转移模型实现预测的目的,实验结果如图38所示。GAN可以用于自动驾驶中的半监督学习或无监督学习任务,还可以利用实际场景不断更新的视频帧来实时优化GAN的生成器。Gou 等^[37]提出利用仿真图像和真实图像作为训练样本来实现人眼检测,但是这种仿真图像与真实图像存在一定的分布差距。Shrivastava等^[38]提出一种基于GAN的方法(称为SimGAN),利用无标签真实图像来丰富细化仿真图像,使得合成图像更加真实。作者引入一个自正则化项来实现最小化合成误差并最大程度保留仿真图像的类别,同时利用加入的局部对抗损失函数来对每个局部图像块进行判别,使得局部信息更加丰富。



图36. 基于GAN的生成GAN的生成图像示例。左图为双三次差值结果，SRGAN结果。



图37. 基于GAN的生成图像示例（奇数列为生成图像，偶数列为目标图像）

5.2 语音和语言领域

目前已经有一些关于GAN 的语音和语言处理文章。Li等^[39]提出用GAN来表征对话之间的隐式关联性,从而生成对话文本。Zhang 等^[40]提出基于GAN 的文本生成,他们用CNN作为判别器,判别器基于拟合LSTM的输出,用矩匹配来解决优化问题;在训练时,和传统更新多次判别器参数再更新一次生成器不同,需要多次更新生成器再更新CNN判别器。SeqGAN基于策略梯度来训练生成器 G ,策略梯度的反馈奖励信号来自于生成器经过蒙特卡洛搜索得到,实验表明SeqGAN在语音、诗词和音乐生成方面可以超过传统方法。Reed等^[41]提出用GAN 基于文本描述来生成图像,文本编码被作为生成器的条件输入,同时为了利用文本编码信息,也将其作为判别器特定层的额外信息输入来改进判别器,判别是否满足文本描述的准确率,实验结果表明生成图像和文本描述具有较高相关性。

5.2.1 个人观点

GAN在图像上的研究已经取得了很大的进步。然而,相比图像来说,文字生成可能会麻烦一些。因为文字的分布不像图像那样,它更复杂,所以捕捉它不会像捕捉图像那么完美。具体原因肯能分为以下几点:

1. 文字分布难以捕捉:图像数据集的分布相比文字而言很一致,比如人头的数据集,它每张照片都是人头,都有特定的五官。如果再人头后面加上不同的背景,试想一下,生成新的图片能否完美地包括新的人头和新的背景吗。
2. 文字难以变动:还以人头为例,如果生成的人头嘴巴倾斜一点或者眼睛倾斜一点,这都不会严重地影响照片的质量,因为它还是一张人头像。但是一段文字对顺序的要求是很高的,所以变动一点,文字就可能不具有语义了。
3. 卷积网络的强大:卷积网络对图像的计算能力是强大的。事实上,像SeqGAN,判别器是用卷积来对文字进行计算的。而像前面所说,有人利用RNN或者LSTM试图生成文字。但是由于前面两种原因,生成文字的效果可能不会像图片那么好。

5.3 其他领域

除了将GAN应用于图像和视觉、语音和语言等领域,GAN还可以与强化学习相结合,例如前述的SeqGAN。还有研究者将GAN和模仿学习融合^[42]、将GAN和Actor-critic 方法结合^[43]。Hu等^[44]提出MalGAN帮助检测恶意代码,用GAN生成具有对抗性的病毒代码样本,实验结果表明基于GAN 的方法可以比传统基于黑盒检测模型的方法性能更好。

6. GAN 的缺陷和发展趋势

GAN虽然解决了生成式模型的一些问题,并且对其他方法的发展具有一定的启发意义,但是GAN并不完美,它在解决已有问题的同时也引入了一些新的问题。GAN 最突出的优点同时

也是它最大的问题根源。GAN采用对抗学习的准则,理论上还不能判断模型的收敛性和均衡点的存在性。训练过程需要保证两个对抗网络的平衡和同步,否则难以得到很好的训练效果。而实际过程中两个对抗网络的同步不易把控,训练过程可能不稳定。另外,作为以神经网络为基础的生成式模型,GAN存在神经网络类模型的一般性缺陷,即可解释性差。另外,GAN生成的样本虽然具有多样性,但是存在崩溃模式现象,可能生成多样的,但对于人类来说差异不大的样本。虽然GAN存在这些问题,但不可否认的是,GAN的研究进展表明它具有广阔的发展前景。例如, WGAN彻底解决了训练不稳定问题,同时基本解决了崩溃模式现象。如何彻底解决崩溃模式并继续优化训练过程是GAN的一个研究方向。另外,关于GAN收敛性和均衡点存在性的理论推断也是未来的一个重要研究课题。以上研究方向是为了更好地解决GAN存在的缺陷。从发展应用GAN的角度,如何根据简单随机的输入,生成多样的、能够与人类交互的数据,是近期的一个应用发展方向。从GAN与其他方法交叉融合的角度,如何将GAN与特征学习、模仿学习、强化学习等技术更好地融合,开发新的人工智能应用或者促进这些方法的发展,是很有意义的发展方向。从长远来看,如何利用GAN推动人工智能的发展与应用,提升人工智能理解世界的能力,甚至激发人工智能的创造力是值得研究者思考的问题。

参考文献

- [1] Goodfellow I, Bengio Y, Courville A. *Deep Learning*. Cambridge, UK: MIT Press, 2016.
- [2] WANG Kun-Feng, Gou Chao, Duan Yan-Jie, Generative Adversarial Networks: The State of the Art and Beyond
- [3] Kingma D P, Welling M. Auto-encoding variational Bayes. arXiv preprint arXiv: 1312.6114, 2013.
- [4] Rezende D J, Mohamed S, Wierstra D. Stochastic back-propagation and approximate inference in deep generative models. arXiv preprint arXiv: 1401.4082, 2014.
- [5] Hinton G E, Sejnowski T J, Ackley D H. Boltzmann Machines: Constraint Satisfaction Networks that Learn. Technical Report No. CMU-CS-84/119, Carnegie-Mellon University, Pittsburgh, PA, USA, 1984.
- [6] Ackley D H, Hinton G E, Sejnowski T J. A learning algorithm for Boltzmann machines. *Cognitive Science*, 1985, 9(1): 147;169
- [7] Bengio Y, Thibodeau-Laufer F, Alain G, Yosinski J. Deep generative stochastic networks trainable by backprop. arXiv preprint arXiv:1306.1091, 2013.
- [8] McDaniel P, Papernot N, Celik Z B. Machine learning in adversarial settings. *IEEE Security & Privacy*, 2016, 14(3):68;72.
- [9] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. *Science*, 2006, 313(5786):504;507
- [10] LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*, 2015, 521(7553): 436;444
- [11] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks. In: Proceedings of the 25th International Conference on Neural Information Processing Systems. Lake Tahoe, Nevada, USA: ACM, 2012. 1097;1105
- [12] He K M, Zhang X Y, Ren S Q, Sun J. Deep residual learning for image recognition. In:

- Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Las Vegas, NV, USA: IEEE, 2016. 770;778.
- [13] Hinton G, Deng L, Yu D, Dahl G E, Mohamed A R, Jaitly N, Senior A, Vanhoucke V, Nguyen P, Sainath T N, Kingsbury B. Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Processing Magazine*, 2012, 29(6): 82;97.
 - [14] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. In: Proceedings of the 2014 Conference on Advances in Neural Information Processing Systems 27. Montreal, Canada: Curran Associates, Inc., 2014. 3104;3112.
 - [15] He D, Chen W, Wang L W, Liu T Y. A game-theoretic machine learning approach for revenue maximization in sponsored search. arXiv preprint arXiv: 1406.0728, 2014.
 - [16] Silver D, Huang A, Maddison C J, Guez A, Sifre L, van Den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, Hassabis D. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016, 529(7587):484;489.
 - [17] Schmidhuber J. Learning factorial codes by predictability minimization. *Neural Computation*, 1992, 4(6): 863;879.
 - [18] Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, Marchand M, Lempitsky V. Domainadversarial training of neural networks. *Journal of Machine Learning Research*, 2016, 17(59): 1;35.
 - [19] Chen W Z, Wang H, Li Y Y, Su H, Wang Z H, Tu C H, Lischinski D, Cohen-Or D, Chen B. Synthesizing training images for boosting human 3D pose estimation. In: Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV). Stanford, CA, USA: IEEE, 2016. 479;488
 - [20] Antonia Greswell, Tom White, Vincent Dumoulin, Kai Arulkumaran.
 - [21] Szegedy C, Zaremba W, Sutskever I, Bruna J, Erhan D, Goodfellow I, Fergus R. Intriguing properties of neural networks. arXiv preprint arXiv: 1312.6199, 2013.
 - [22] Alec R, Luke M, Soumith C, Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:1511.06434, 2015.
 - [23] J. Gauthier. Conditional generative adversarial nets for convolutional face generation. Class Project for Stanford CS231N:Convolutional Neural Networks for Visual Recognition, Winter semester 2014
 - [24] Sergey L, Christian S. Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv:1502.03167
 - [25] Arjovsky M, Chintala S, Bottou L. Wasserstein GAN. arXiv preprint arXiv: 1701.07875, 2017.
 - [26] Ishaan G, Faruk A, Martin A, Vincent D, Improved Training of Wasserstein GANs. arXiv:1704.00028v3.
 - [27] Mirza M, Osindero S. Conditional generative adversarial nets. arXiv preprint arXiv: 1411.1784, 2014.
 - [28] Pfau D, Vinyals O. Connecting generative adversarial networks and actor-critic methods. arXiv preprint arXiv:1610.01945, 2016.
 - [29] Emily D, Soumith C, Arthur S, Rob F. Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks. aeXiv;1506.05751
 - [30] Chen X, Duan Y, Houthoofd R, Schulman J, Sutskever I, Abbeel P. InfoGAN: interpretable representation learning by information maximizing

- generative adversarial nets. In: Proceedings of the 2016 Neural Information Processing Systems. Barcelona, Spain: Department of Information Technology IMEC, 2016. 2172;2180
- [31] Yu L T, Zhang W N, Wang J, Yu Y. SeqGAN: sequence generative adversarial nets with policy gradient. arXiv preprint arXiv: 1609.05473, 2016.
 - [32] Qi G J. Loss-sensitive generative adversarial networks on Lipschitz densities. arXiv preprint arXiv: 1701.06264, 2017.
 - [33] Odena A. Semi-supervised learning with generative adversarial networks. arXiv preprint arXiv: 1606.01583, 2016.
 - [34] Donahue J, Krähenbühl P, Darrell T. Adversarial feature learning. arXiv preprint arXiv:1605.09782, 2016.
 - [35] Odena A, Olah C, Shlens J. Conditional image synthesis with auxiliary classifier GANs. arXiv preprint arXiv:1610.09585, 2016.
 - [36] Mario L, Karol K, Marcin M, Sylvain G, Oliver B, Google Brain. Are GANs Created Equal? A large-Scale Study.
 - [37] Gou C, Wu Y, Wang K, Wang F Y, Ji Q. Learning-by-synthesis for accurate eye detection. In: Proceedings of the 2016 IEEE International Conference on Pattern Recognition (ICPR). Cancun, Mexico: IEEE, 2016.
 - [38] Shrivastava A, Pfister T, Tuzel O, Susskind J, Wang W D, Webb R. Learning from simulated and unsupervised images through adversarial training. arXiv preprint arXiv:1612.07828, 2016.
 - [39] Li J W, Monroe W, Shi T L, Jean S, Ritter A, Jurafsky D. Adversarial learning for neural dialogue generation. arXiv preprint arXiv:1701.06547, 2017.
 - [40] Yu L T, Zhang W N, Wang J, Yu Y. SeqGAN: sequence generative adversarial nets with policy gradient. arXiv preprint arXiv: 1609.05473, 2016.
 - [41] Reed S, Akata Z, Yan X C, Logeswaran L, Lee H, Schiele B. Generative adversarial text to image synthesis. In: Proceedings of the 33rd International Conference on Machine Learning. New York, NY, USA: ICML, 2016.
 - [42] Ho J, Ermon S. Generative adversarial imitation learning. In: Proceedings of the 2016 Conference on Advances in Neural Information Processing Systems 29. Curran Associates, Inc., 2016. 4565;4573.
 - [43] Finn C, Christiano P, Abbeel P, Levine S. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. arXiv preprint arXiv:1611.03852, 2016.
 - [44] Pfau D, Vinyals O. Connecting generative adversarial networks and actor-critic methods. arXiv preprint arXiv:1610.01945, 2016.

