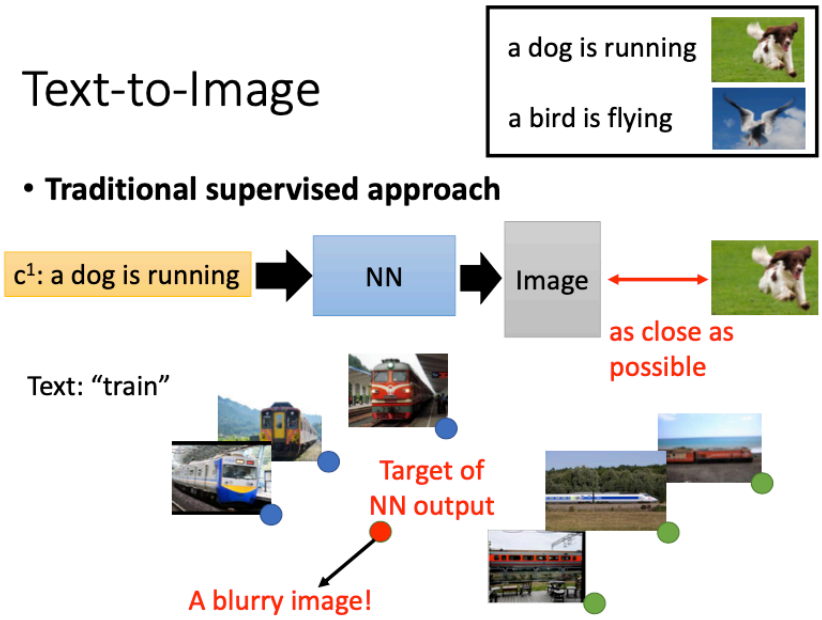
对于一般的GAN，都是随机生成一个vector，然后输入GAN，再生成一张image，但我们并不能控制output。对于本文要讲述的CGAN，我们则可以操控其输出的结果。

Text-to-Image

对于传统的监督学习的方法，训练数据集是一些带有描述的图片，网络的input是一段文字，output则是一张图片，我们希望输出的图片和target越接近越好。



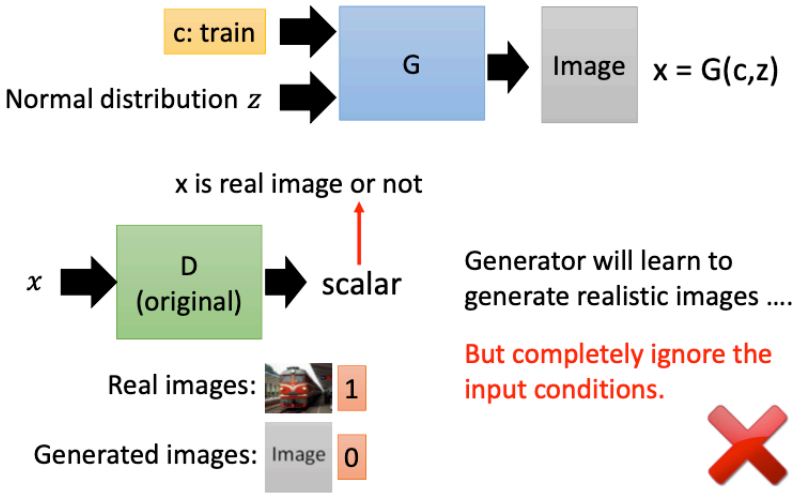
如果现在网络的输入文字是“train”，网络会觉得正面的火车是对的，侧面的火车也是对的，网络最后会取这些值的一个平均值，因此会得到一个非常模糊的火车图片。

Conditional GAN

现在我们使用CGAN来完成这个任务。CGAN现在的generator的输入不仅有z，还有一个condition c（“train”），G的输出为$x = G(z, c)$；

这时我们还使用原来的Discriminator，把x输入D，D可以对其进行评价分数scalar，对真实的图像输出为1，生成的图像输出为0。

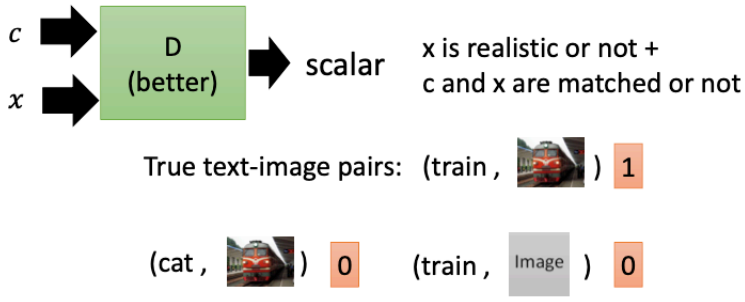
那么现在就出现了一个新问题，G可以完全不管输入的condition，只生成高质量的、接近真实的图像即可。比如我们把condition设置为“dog”，G如果生成一只猫的图像，这个图像很接近真实图像，那么就可以骗过discriminator。



因此，这里的discriminator也要做出改变，需要把condition也作为D的其中一个输入。

那么现在D输出的分数就有两部分组成：（1）x的真实性；（2）x是不是满足condition的条件。如果这个图片和文字是match的，而且图片很接近真实图像，D就会给这个图像一个高分。

给低分0的两个case：（1）如果给出了正确的文字，但生成了模糊的图像；（2）虽然生成了清晰的图像，但和随机输入的文字（condition）是不匹配的。



这里是具体的算法。

Learning D：输入是文字和图像的pair，即$\{(c^1,x^1),\dots,(c^m,x^m)\}$，从（高斯）分布中取出noise $\{z^1,z^2\dots,z^m\}$，输入G，得到生成的图像 $\tilde{x}^i=G(c^i,z^i)$；从database中取出m个真实的图像数据 $\{\hat{x}^1,\dots,\hat{x}^m\}$，再输入discriminator D，不断调整$\theta_d$，使得得到的分数越大越好，

$$\tilde{V}=\frac{1}{m}\sum_{i=1}^{m}logD(c^i,x^i)+\frac{1}{m}\sum_{i=1}^{m}log(1-D(c^i,\tilde{x}^i))+\frac{1}{m}\sum_{i=1}^{m}log(1-D(c^i,\hat{x}^i))$$

其中$D(c^i,x^i)$表示真实图像所得到的分数，D的目标就是使真实图像获得的分数越大越好；而$D(c^i,\tilde{x}^i)$表示G生成的图像所得到的分数，应该越小越好，所以前面加了负号；而$D(c^i,\hat{x}^i)$表示生成了清晰的图像，但和condition不匹配，所以前面加了负号。后两个case都是给低分的情况。

其他过程和之前的GAN差别不大，主要是计算低分数的情况多了一个case。

- In each training iteration:

Learning D

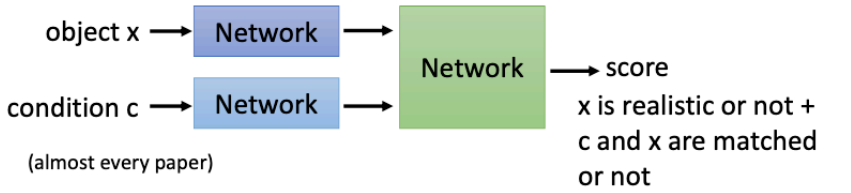
- Sample m positive examples $\{(c^1, x^1), (c^2, x^2), \dots, (c^m, x^m)\}$ from database
- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from a distribution
- Obtaining generated data $\{\tilde{x}^1, \tilde{x}^2, \dots, \tilde{x}^m\}$, $\tilde{x}^i = G(c^i, z^i)$
- Sample m objects $\{\hat{x}^1, \hat{x}^2, \dots, \hat{x}^m\}$ from database
- Update discriminator parameters θ_d to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log D(c^i, x^i)$
    $+\frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(c^i, \tilde{x}^i)\right) + \frac{1}{m}\sum_{i=1}^{m} log\left(1 - D(c^i, \hat{x}^i)\right)$
  - $\theta_d \leftarrow \theta_d + \eta \nabla \tilde{V}(\theta_d)$

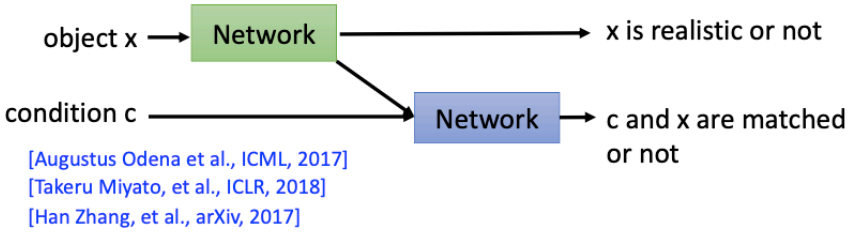
Learning G

- Sample m noise samples $\{z^1, z^2, \dots, z^m\}$ from a distribution
- Sample m conditions $\{c^1, c^2, \dots, c^m\}$ from a database
- Update generator parameters θ_g to maximize
  - $\tilde{V} = \frac{1}{m}\sum_{i=1}^{m} log\left(D\left(G(c^i, z^i)\right)\right)$, $\theta_g \leftarrow \theta_g - \eta \nabla \tilde{V}(\theta_g)$

Conditional GAN - Discriminator

对于一般的discriminator架构，输入为图像和文字，输出为两部分的分数（x的真实性、x和condition是不是match）。如果这个图片和文字是match的，而且图片很接近真实图像，D就会给这个图像一个高分。





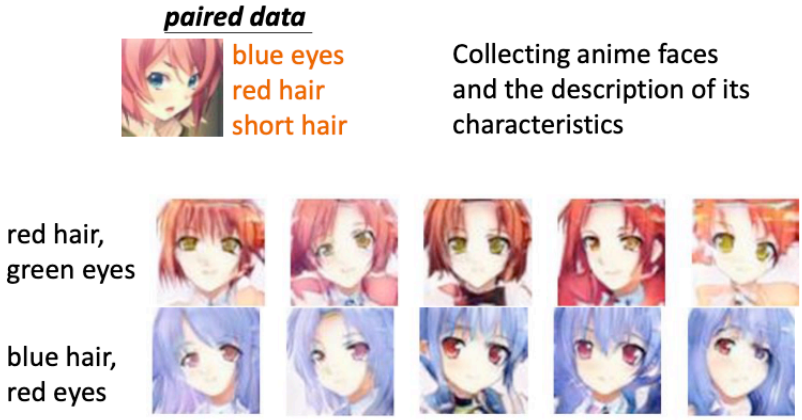
[Augustus Odena et al., ICML, 2017]
[Takeru Miyato, et al., ICLR, 2018]
[Han Zhang, et al., arXiv, 2017]

有学者在其他论文中也提出了其他架构，而且效果还不错。首先有object输入network，得出第一个分数，表示x是否真实；network还会输出一个embedding，再和condition结合，输入network后得到另外一个分数，表示x和c是否match。

选择第一种架构方法有一个缺点。前文我们提到CGAN会给低分的两种case：x和condition是match的，但生成的图像不好；x和condition是不match的，但生成的图像质量高。如果我们使用第一种架构，网络会比较confused，网络并不知道分数低的原因到底是哪一种case，有可能是图像不够realistic，也有可能是和condition不够match。但对于第二种架构，网络则可以清楚地知道到底是因为

哪个原因导致的低分。

下面是结果的展示，输入文字condition，输出对应的图像。



Stack GAN

首先输入一段文字，这些文字先经过embedding的过程，再经过conditioning augmentation的过程（加入噪声），输入generator之后，会生成一张图像（64×64）；discriminator再来判断这个生成的image和输入的文字是不是match的；

这里还有第二个generator，输入为刚才64×64的图像和文字的embedding结果；再生成一张256×256的图像；再把新生成的图像输入discriminator，看到底是不是realistic。

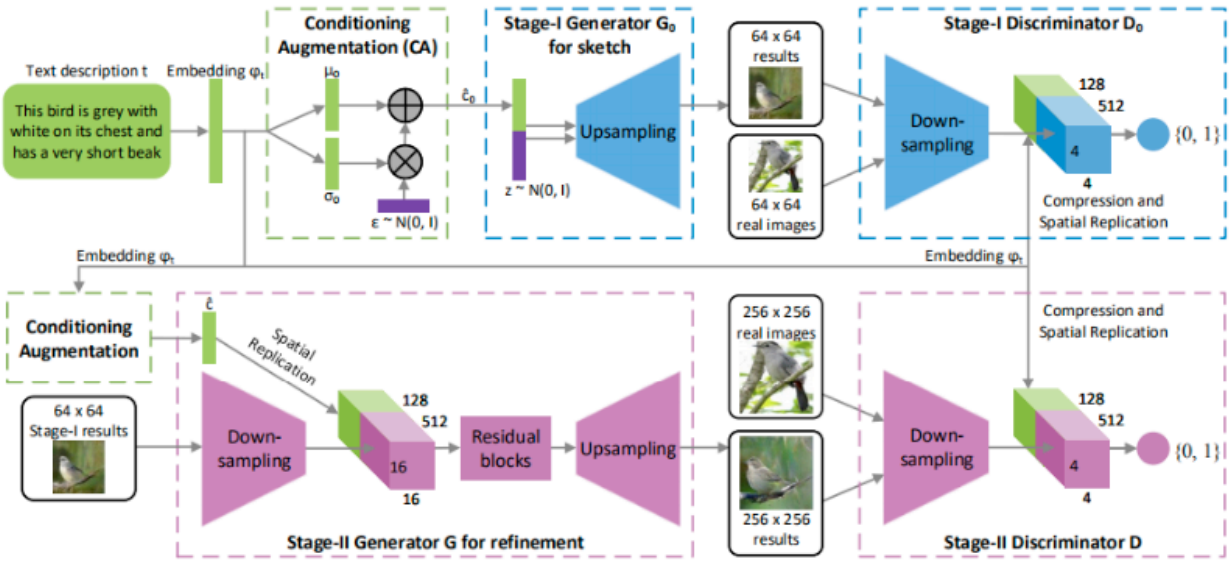
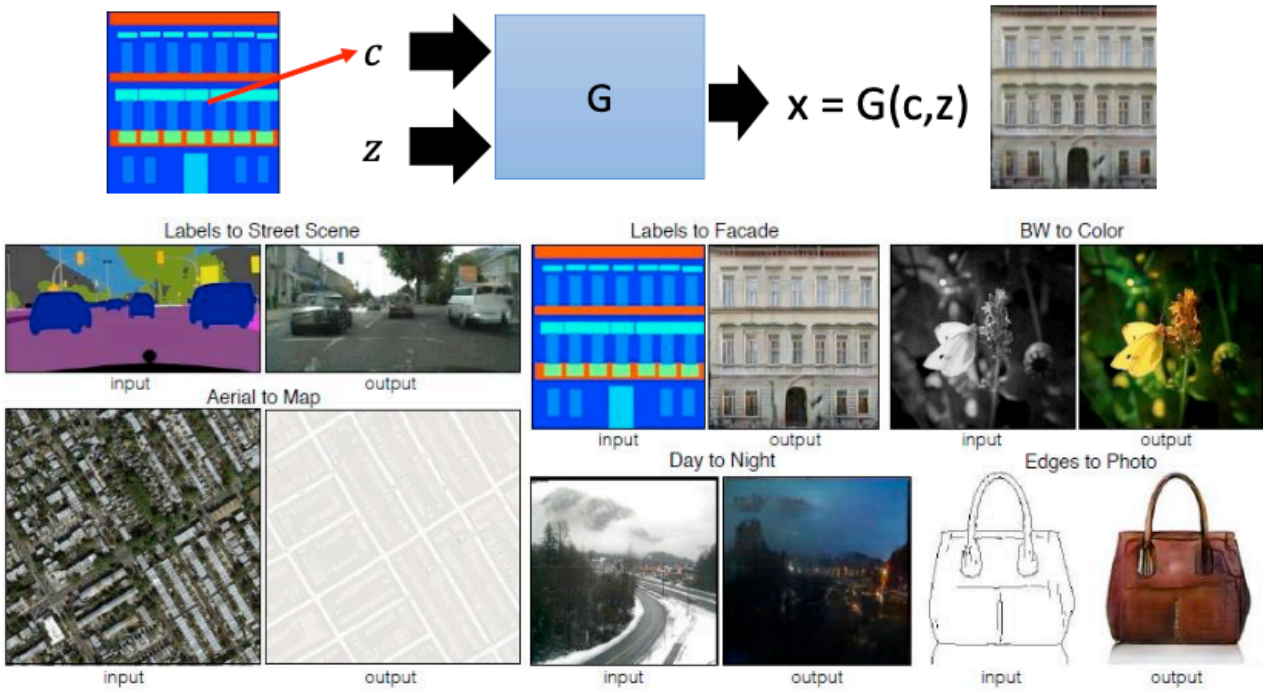


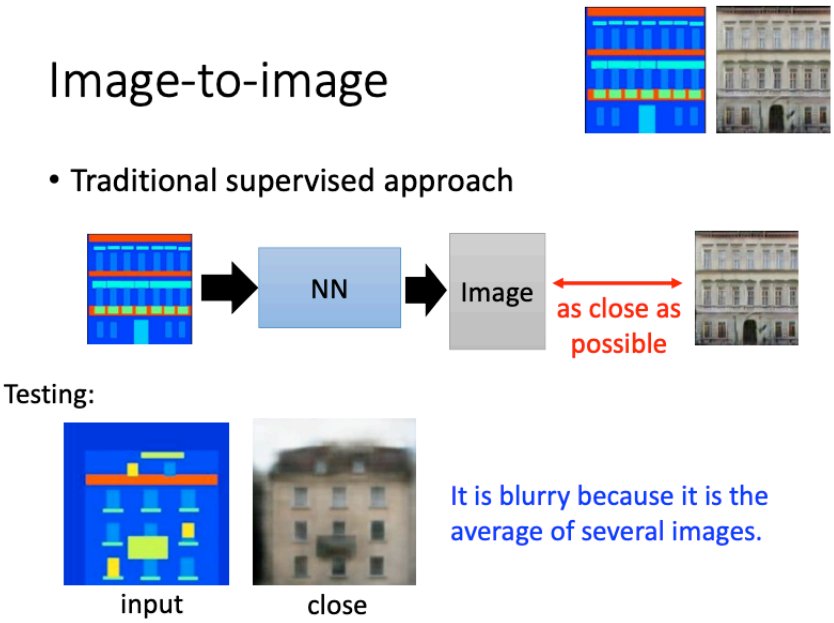
Image-to-image

Introduction

CGAN不仅可以输入一段文字产生一张图像，也可以是image-to-image，比如把简单的几何模型转化为真实的房屋模型，把灰度图转化为彩色图，



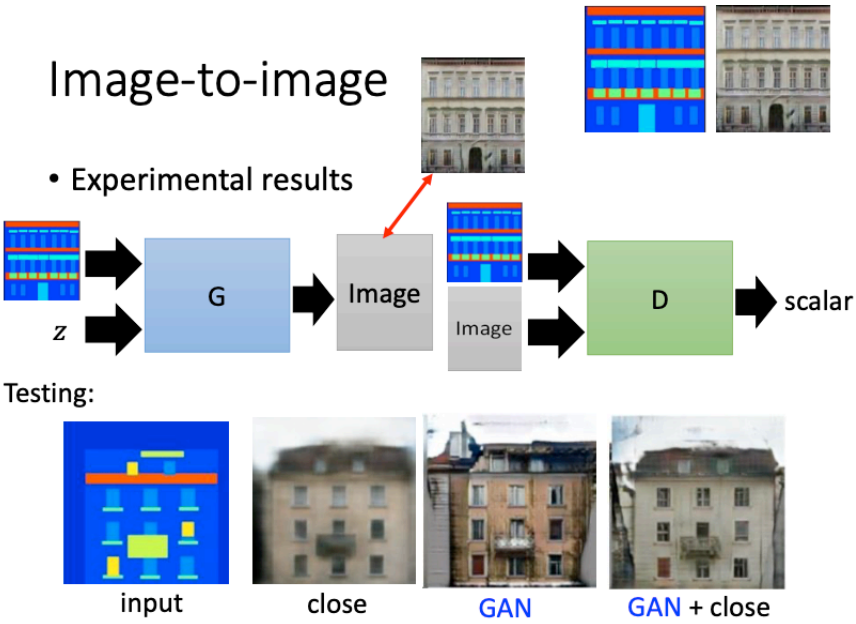
如果使用传统的监督学习的方法来完成这个任务，首先需要收集训练数据（几何模型图、对应的真实房屋图），训练network，output是一张图片，我们希望输出的图片和target越接近越好。由于network会产生多种多样的房子，最后的结果会取一个平均值，因此会产生一个非常模糊的图像。



我们可以使用GAN来完成这个任务。首先把从distribution中sample出来的z和结构图作为G的输入，G会生成一张新的Image；再把Image和原来的结构图（一个pair）输入D，会输出一个分数scalar。可以发现GAN生成的图要相对清晰很多。

但我们这时发现了一个新问题，GAN产生了一些原来的结构图中没有的东西，比如在图的左上角，产生了一个像是窗户或者天线的东西。这时我们可以加入一个新的constrain（真实的房屋图），希望generator产生的image和训练数据集中对应的图像也越靠近越好。

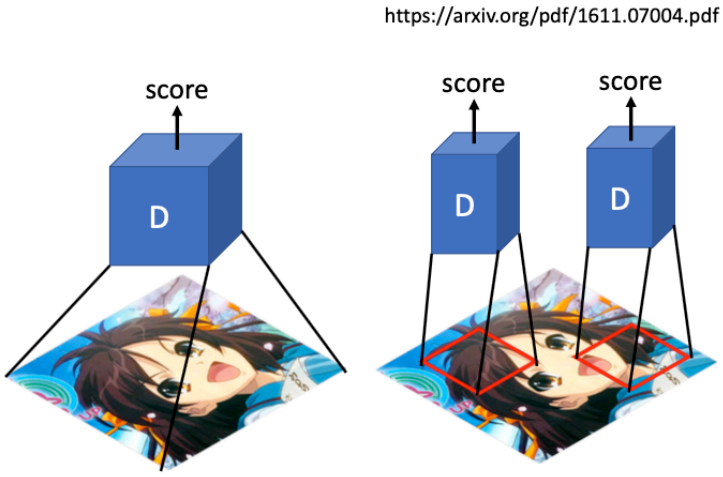
这时generator的目标就有两个：产生出足够清晰可以骗过D的图像；产生的新图像和原来的target要接近。这样就会产生结果很好的图（GAN+close），图足够清晰，也不会产生一些奇怪的东西。



Patch GAN

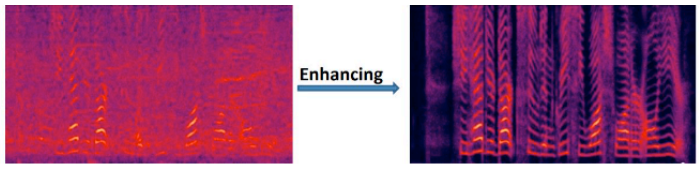
现在要生成的是一张很大的图像，如果这时discriminator的输入还是一整张大的图像，那么D要对其进行评分，就肯定需要更多的参数，很有可能产生overfiting或者训练所需的时间会很长。

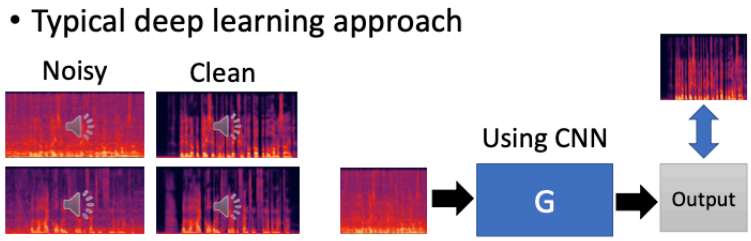
在这篇论文中，作者也对discriminator的设计进行了变化。discriminator只需要检查图中的一小部分，对这一小块图片来输出评价分数 。区域的大小也是需要调整的参数之一。



Speech Enhancement

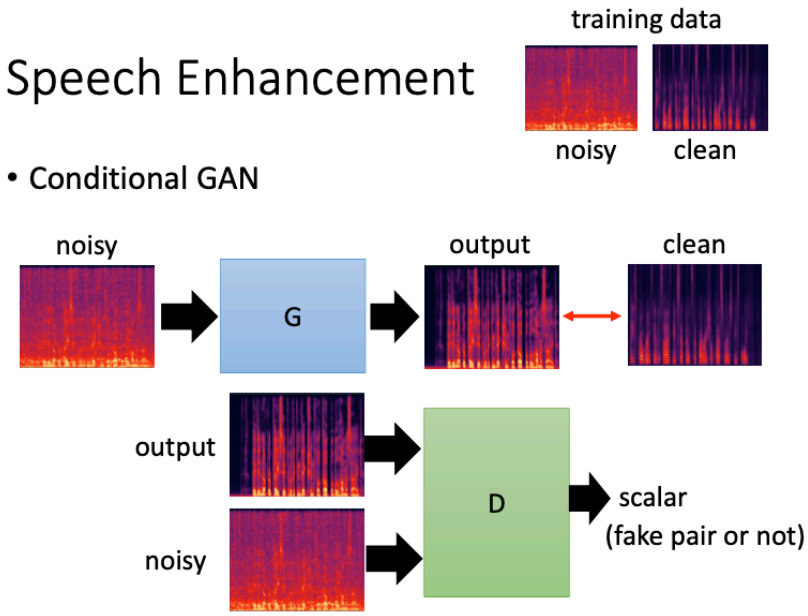
如果我们使用传统的深度学习方法来做语音增强，首先需要把纯净语音加上一些noise再输入CNN，不断地训练CNN，使其能够输出去噪后的纯净语音。





直接产生同样会有语音频谱图比较模糊的情况，我们在这里也可以使用CGAN算法。

输入为带noise的语音信号，G的输出为增强语音，增强语音和纯净语音之间应该越接近越好。discriminator的输入为增强语音和带噪语音，输出评价分数，看这个output是不是clean的，还要看output和noise这个pair是不是match的。



Video Generation

输入一段video，让generator预测下一步会发生什么，产生对应的video；discriminator要同时考虑generator的input和output，可以把它们接到一起，变成一段完整的影片，让discriminator来判断到底是不是一个合理的影片。

Video Generation