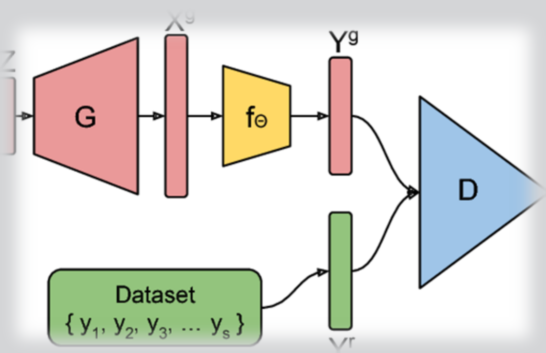


AmbientGAN: Generative models from lossy measurements

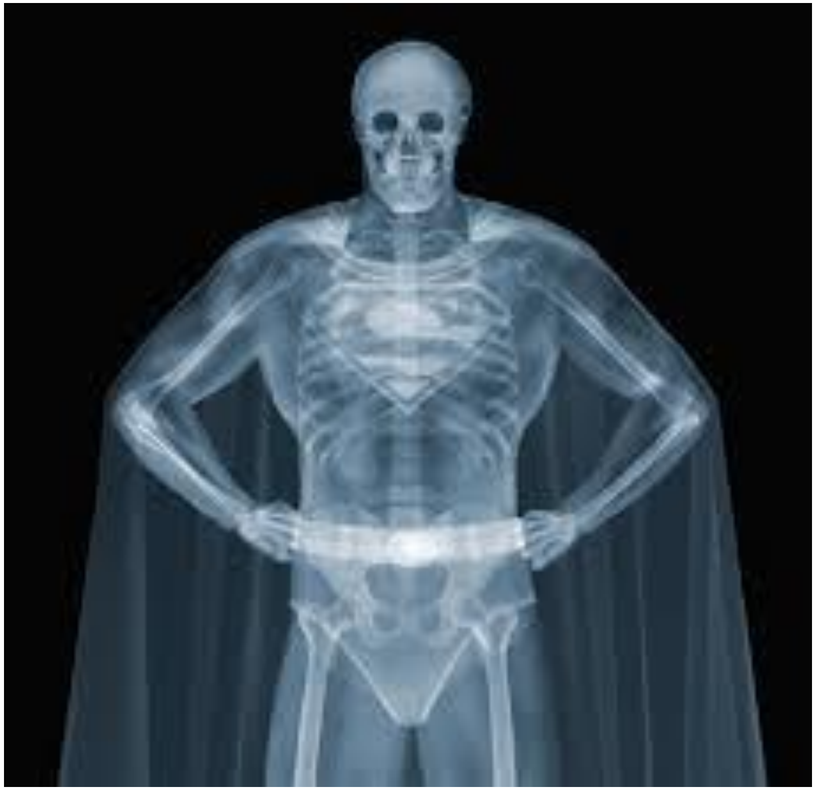
Motivation

"How can we train our generative models with partial, noisy observations?"

Why do we care?

In many settings, it is expensive or even impossible to obtain fully-observed samples, but economical to obtain partial, noisy samples

Example



$$\begin{pmatrix}
1.0 & 0 & 5.0 & 0 & 0 & 0 & 0 & 0 \\
0 & 3.0 & 0 & 0 & 0 & 0 & 11.0 & 0 \\
0 & 0 & 0 & 0 & 9.0 & 0 & 0 & 0 \\
0 & 0 & 6.0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 7.0 & 0 & 0 & 0 & 0 \\
2.0 & 0 & 0 & 0 & 0 & 10.0 & 0 & 0 \\
0 & 0 & 0 & 8.0 & 0 & 0 & 0 & 0 \\
0 & 4.0 & 0 & 0 & 0 & 0 & 0 & 12.0
\end{pmatrix}$$

This paper proposes

- **AmbientGAN:** train the discriminator **not on the raw data domain but on the measurement domain**

- Propose the method to train the generative model with **a noisy, corrupted, or missing data**

- Prove that it is **theoretically possible to recover the original true data distribution** even though the measurement process is **<u>not invertible</u>**

This paper proposes (preview)

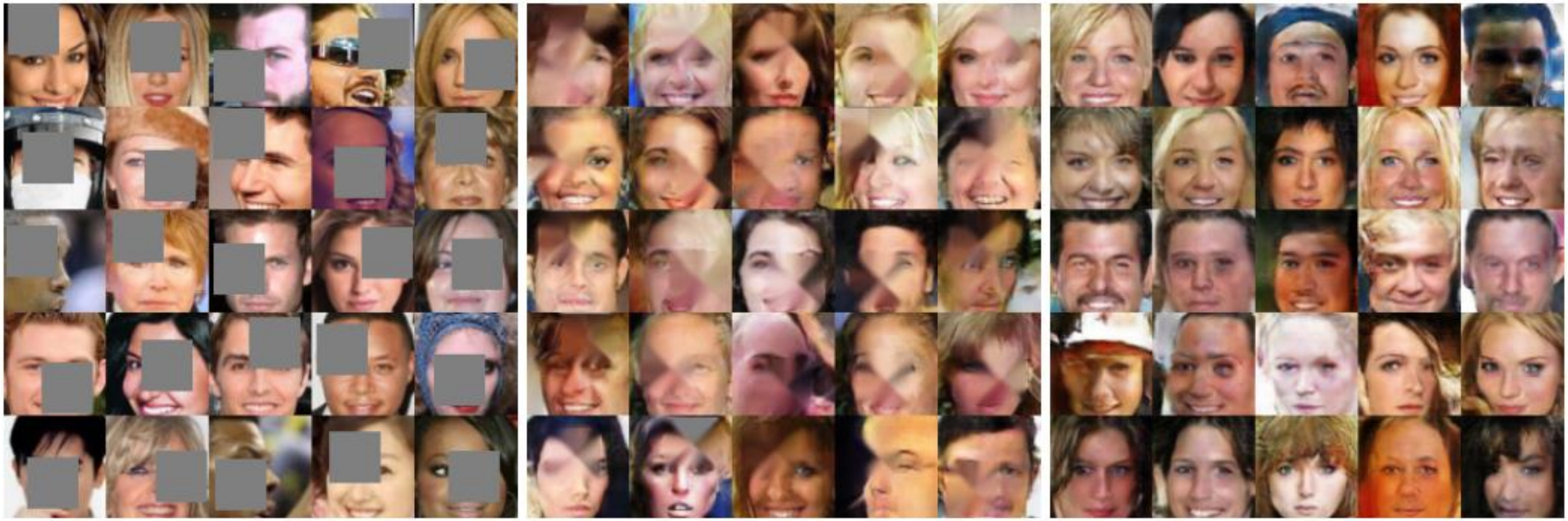
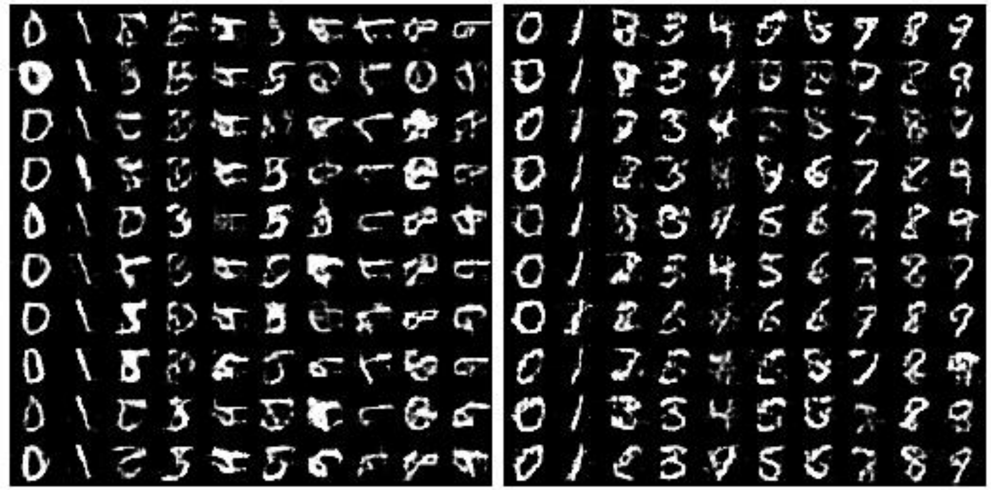


Figure 2: (Left) Samples of lossy measurements used for training. Samples produced by (middle) a baseline that trains from inpainted images, and (right) our model.

This paper proposes (preview)

(a) (left) Samples of lossy measurements. Each image is a blurred noisy version of the original. Samples produced by (middle) a baseline that uses Wiener deconvolution, and (right) our model.

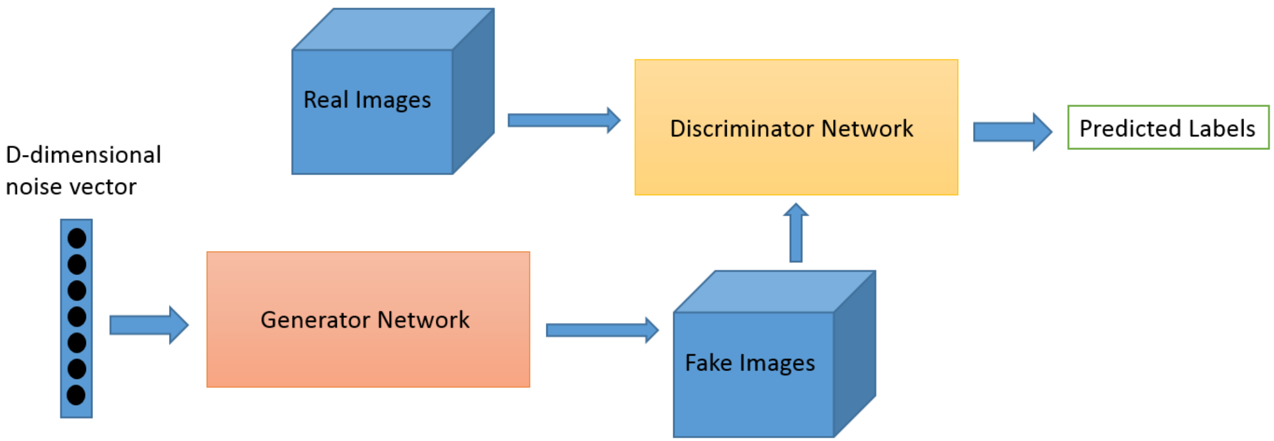


(b) Samples produced by our model trained from two 1D projections of each image. On left, the training data does not include the angle of the projections, so it cannot identify orientation or chirality. On right, the training data includes the angle.

Generative Adversarial Nets (review)

- **Diagram of Standard GAN**

- Generative Adversarial Nets (GANs) are composed of two components
- The main idea behind a GANs are to have two competing neural network model
  - Generator(G): to create natural looking images that are similar to the original data distribution
  - Discriminator(D): determining whether a given image looks natural or looks like it has been artificially created.



<Frameworks of GANs>

Generative Adversarial Nets (review)

- **Diagram of Standard GAN**

- Generative Adversarial Nets (GANs) are composed of two components
- The main idea behind a GANs are to have two competing neural network model
  - Generator(G): to create natural looking images that are similar to the original data distribution
  - Discriminator(D): determining whether a given image looks natural or looks like it has been artificially created.

$$\min_{G} \max_{D} V(D,G) = E_{x\sim p_{data}(x)}[logD(x)] + E_{z\sim p_{z}(z)}\left[\log\left(1 - D(G(z))\right)\right]$$

<Objective function of GANs>

Generative Adversarial Nets (review)

- **Diagram of Standard GAN**

- Generative Adversarial Nets (GANs) are composed of two components
- The main idea behind a GANs are to have two competing neural network model
  - Generator(G): to create natural looking images that are similar to the original data distribution
  - Discriminator(D): determining whether a given image looks natural or looks like it has been artificially created.

Limitation of GAN

- **Training Stability**
- **Forgetting problem**
- **Requires Good (of fully observed) Training samples**

Limitation of GAN

- **Training Stability**

- Forgetting problem
- Requires Good (of fully observed) Training samples

Limitation of GAN

- Training Stability

- **Forgetting problem**

- Requires Good (of fully observed) Training samples

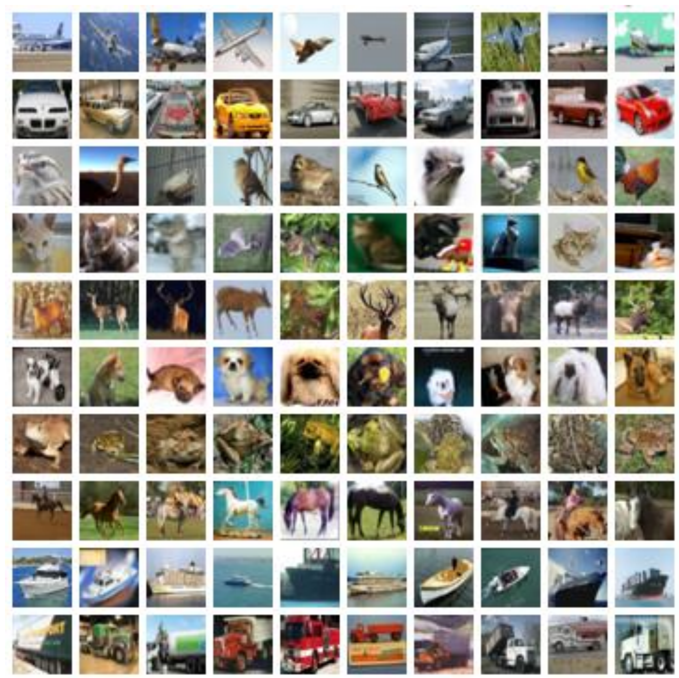
Limitation of GAN

- Training Stability
- Forgetting problem
- **Requires Good (of fully observed) Training samples**
  - To train the generator, we need a lot of good images
  - In many settings, it is expensive or even impossible to obtain **fully-observed samples**, but economical to obtain **partial**, **noisy** samples

Limitation of GAN

- Training Stability
- Forgetting problem
- **Requires Good (of fully observed) Training samples**

[CelebFaces]



[cifer-10]

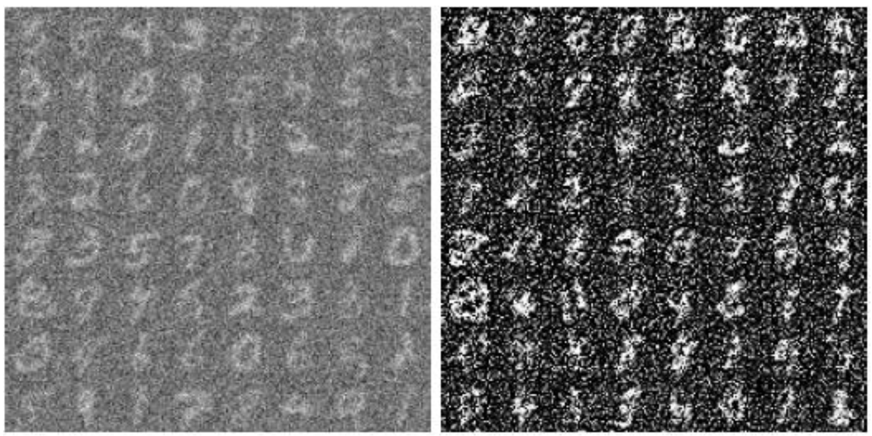
Limitation of GAN

- Training Stability

- Forgetting problem

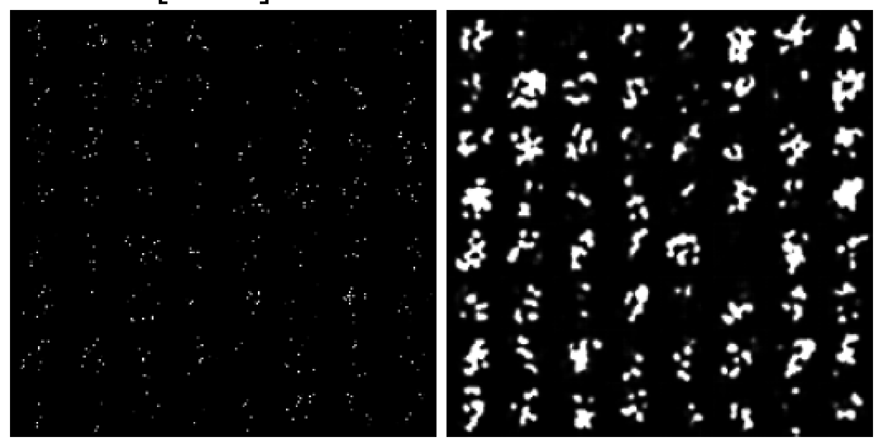
- **Requires Good (of fully observed) Training samples**

[Real] [Fake]



[Noise]

[Real] [Fake]



[Block-pixels]

AmbientGAN framework

Fake Measurement Generative Process

$$X_g = G(Z) \qquad Z \sim p_z$$

$$\Theta \sim p_\theta$$

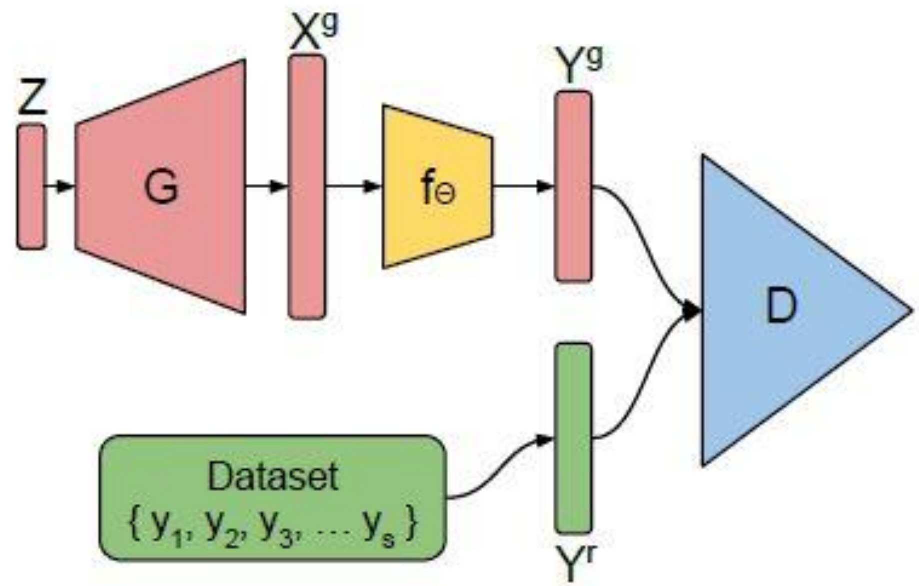
$$Y_g = f_\theta(X_g) = f_\theta(G(Z))$$

Real Measurement Generative Process

$$X \sim p_x^r$$

$$\Theta \sim p_\theta$$

$$Y = f_\theta(X) \sim p_y^r$$



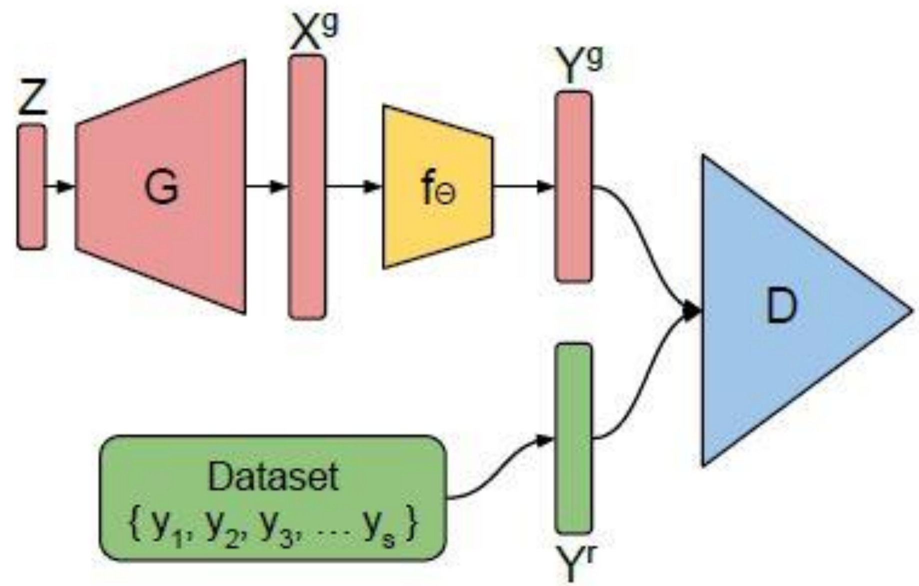
AmbientGAN framework

Fake Measurement Generative Process

$$X_g = G(Z) \qquad Z \sim p_z$$

$$\Theta \sim p_\theta$$

$$Y_g = f_\theta(X_g) = f_\theta(G(Z))$$



Real Measurement Generative Process

$$X \sim p_x^r$$

$$\Theta \sim p_\theta$$

$$Y = f_\theta(X) \sim p_y^r$$

$$\min_G \max_D E_{Y^r \sim p_y^r}[(D(Y^r))] + E_{Z \sim p_z, \Theta \sim p_\theta}\left[\left(1 - D\left(f_\theta(G(Z))\right)\right)\right]$$

$$Z \sim p_z \;, \Theta \sim p_\theta, and\ Y^r \sim Unif\{y_1, y_2, \dots, y_s\}$$

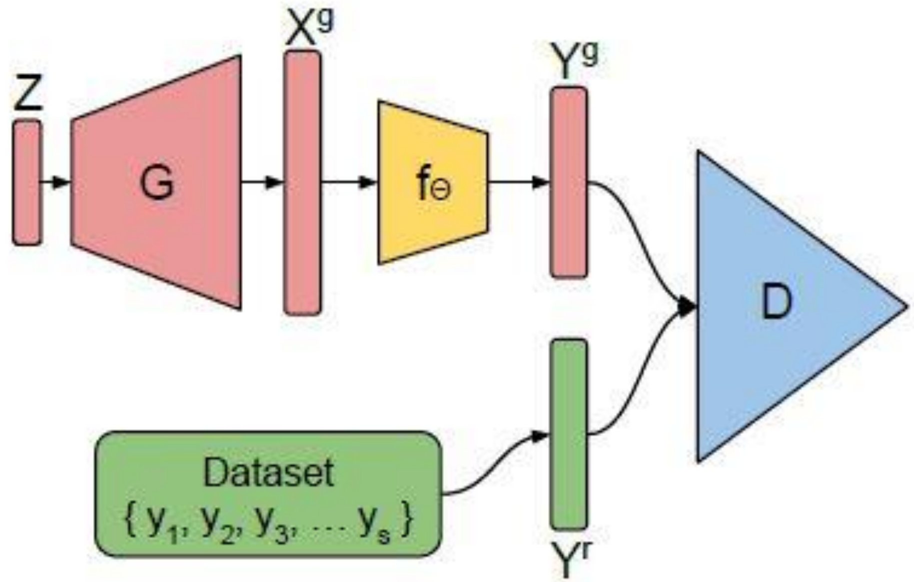
AmbientGAN framework

Fake Measurement Generative Process

$$X_g = G(Z) \qquad Z \sim p_z$$

$$\Theta \sim p_\theta$$

$$Y_g = f_\theta(X_g) = f_\theta(G(Z))$$



Real Measurement Generative Process

$$X \sim p_x^r$$

$$\Theta \sim p_\theta$$

$$Y = f_\theta(X) \sim p_y^r$$

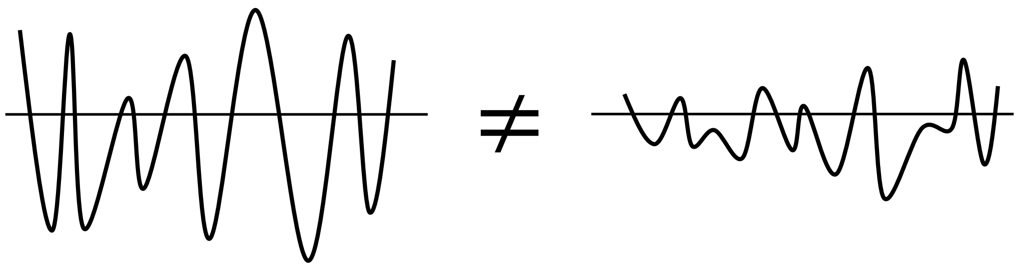
$$\min_G \max_D E_{Y^r \sim p_y^r}[(D(Y^r))] + E_{Z \sim p_z, \Theta \sim p_\theta}\left[\left(1 - D\left(f_\theta(G(Z))\right)\right)\right]$$

$$Z \sim p_z \ , \Theta \sim p_\theta, and\ Y^r \sim Unif\{y_1, y_2, \dots, y_s\}$$

Measurement

- **Standard Compressed Sensing**
  "Compressible" = "Sparse"

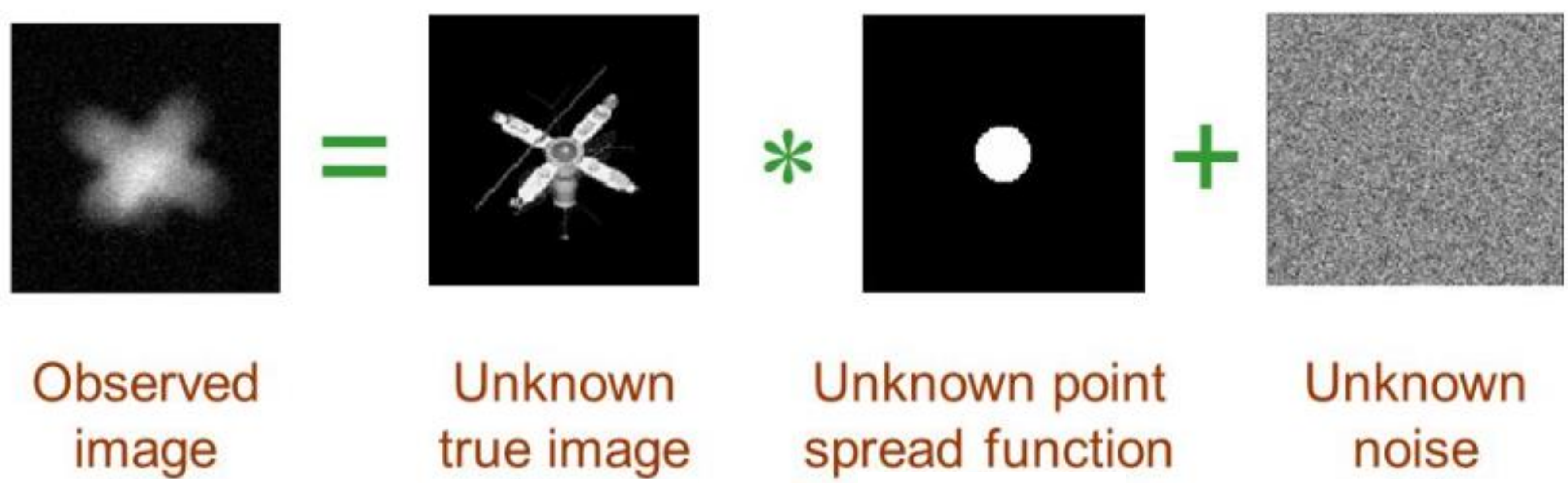
- **Want to estimate x from $y = Ax + \eta$, for $A \in R^{m \times n}$**



[원 신호] [측정 신호]

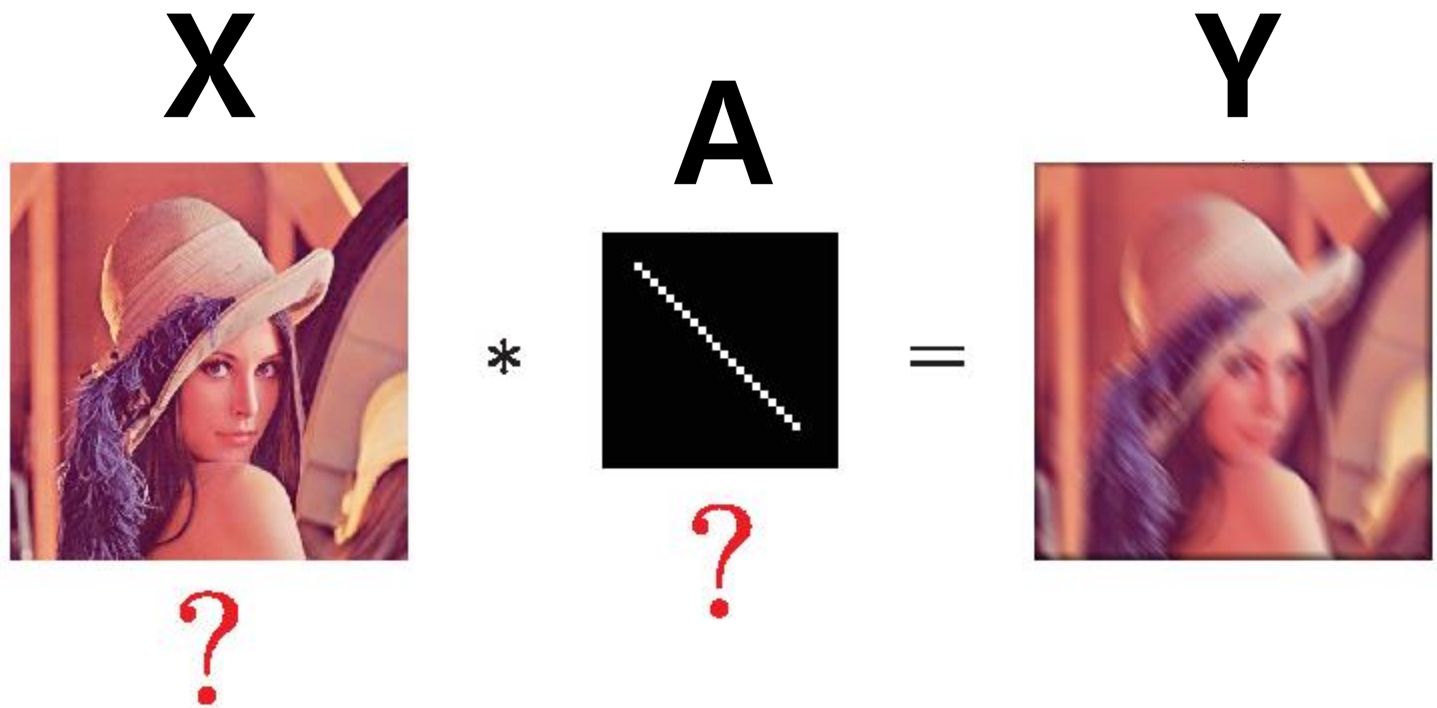
$$\mathbf{X} \xleftarrow{\boldsymbol{A}} \mathbf{Y}$$

Measurement

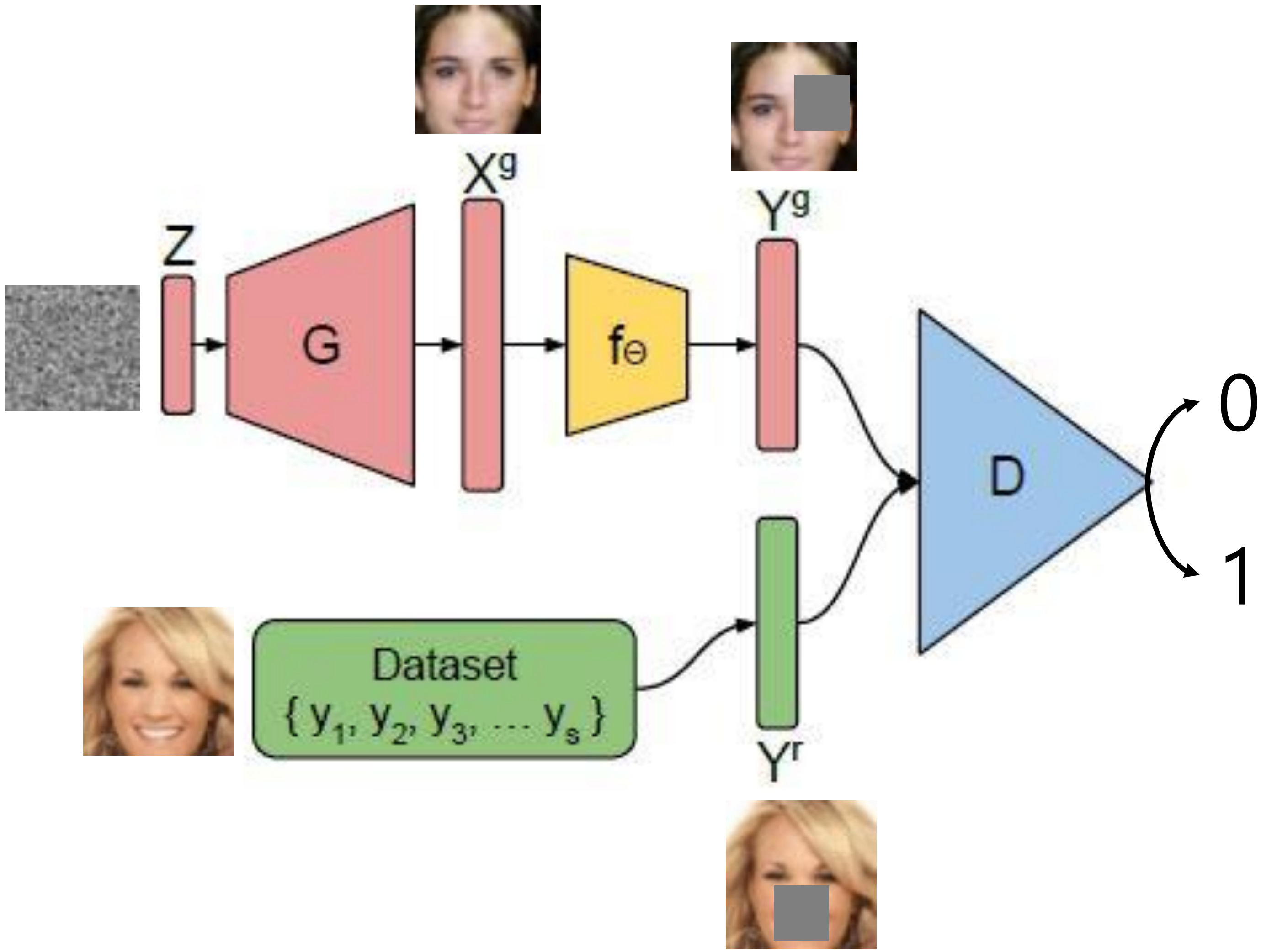


Y **X** **A** $\boldsymbol{\eta}$

Measurement



AmbientGAN



Measurement function

- **Block-Pixels**: with p, make image pixel 0
- **Convolve + Noise**: Gaussian kernel blur + add random Gaussian noise
- **Block-Patch**: make $k \times k$ patch 0
- **Keep-Patch**: make 0 except $k \times k$ patch
- **Extract-Patch**: use $k \times k$ patch as input
- **Pad-Rotate-Project**: zero padding + rotate + vertical 1D projection
- **Pad-Rotate-Project-$\boldsymbol{\theta}$**: **Pad-Rotate-Project** and include θ as an additional information for the measurement

Dataset and Model Architectures

- **MNIST**
  - **Conditional DCGAN**
  - **WGAN**

- **CelebA**
  - **DCGAN**

- **CIFAR-10**
  - **ACGAN**

- For 2D measurement, Block-Pixels, Block-Patch, Keep-Patch, Extract-Patch and Convolve + Noise, conventional Discriminator is used

- For 1D measurement, Pad-Rotate-Project, Pad-Rotate-theta, fully connected Discriminator is used

Baselines

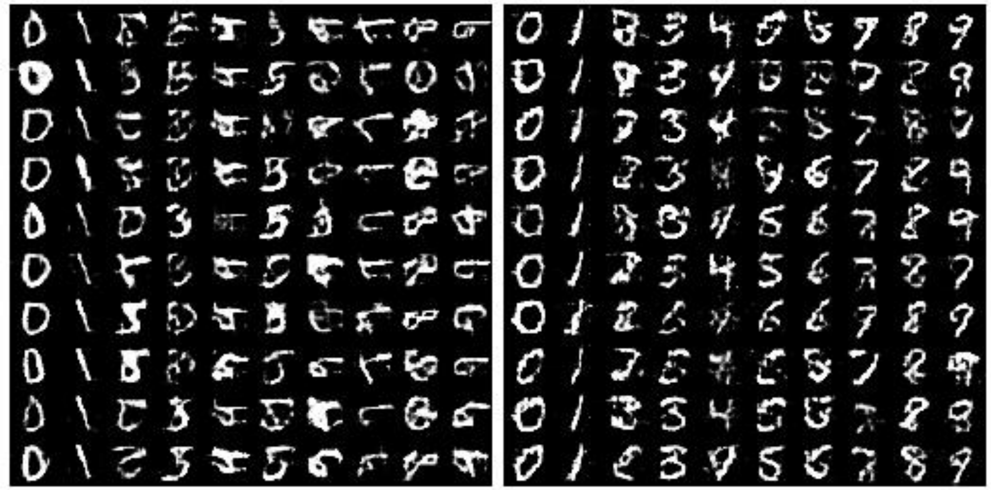
- IGNORE: Learn generative model based on raw measurement
- Unmeasure: Trying to recover the measurements with conventional algorithm
    - Block-Pixels – blur the pixel to fill the zero pixel
    - Convolve+Noise – Winder deconvolution Method
    - Block-Patch – Navier Stroke based inpainting method to fill the zero pixel

Experimental result

Figure 2: (Left) Samples of lossy measurements used for training. Samples produced by (middle) a baseline that trains from inpainted images, and (right) our model.

Experimental result

(a) (left) Samples of lossy measurements. Each image is a blurred noisy version of the original. Samples produced by (middle) a baseline that uses Wiener deconvolution, and (right) our model.



(b) Samples produced by our model trained from two 1D projections of each image. On left, the training data does not include the angle of the projections, so it cannot identify orientation or chirality. On right, the training data includes the angle.

Experimental result

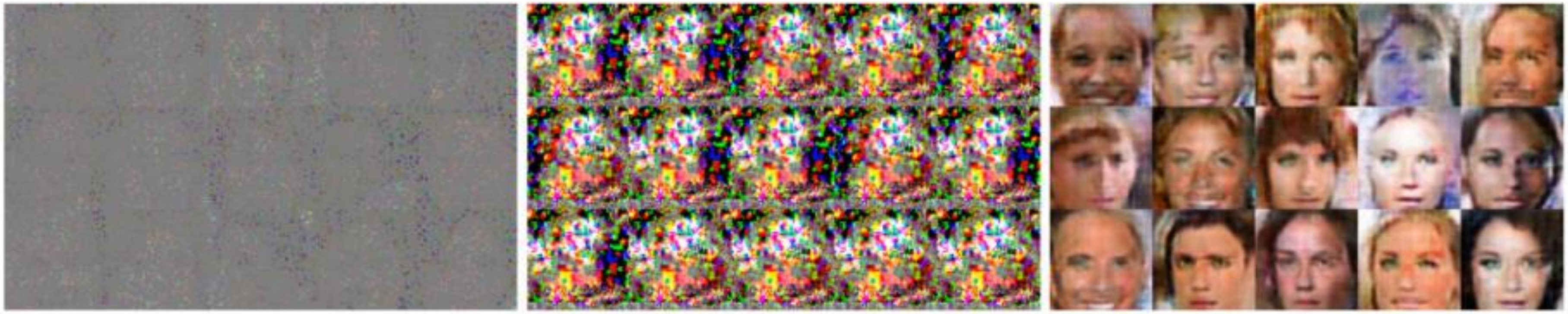


Figure 4: Results with Block-Pixels on celebA. (left) Samples of lossy measurements. Each pixel is blocked independently with probability $p = 0.95$. Samples produced by (middle) unmeasure-blur baseline, and (right) our model.

(a) (left) Samples of lossy measurements. All except a randomly chosen 32×32 patch is set to zero. (right) Samples produced by our model.

(b) Samples produced by our model with Pad-Rotate-Project-θ measurements.

Experimental result

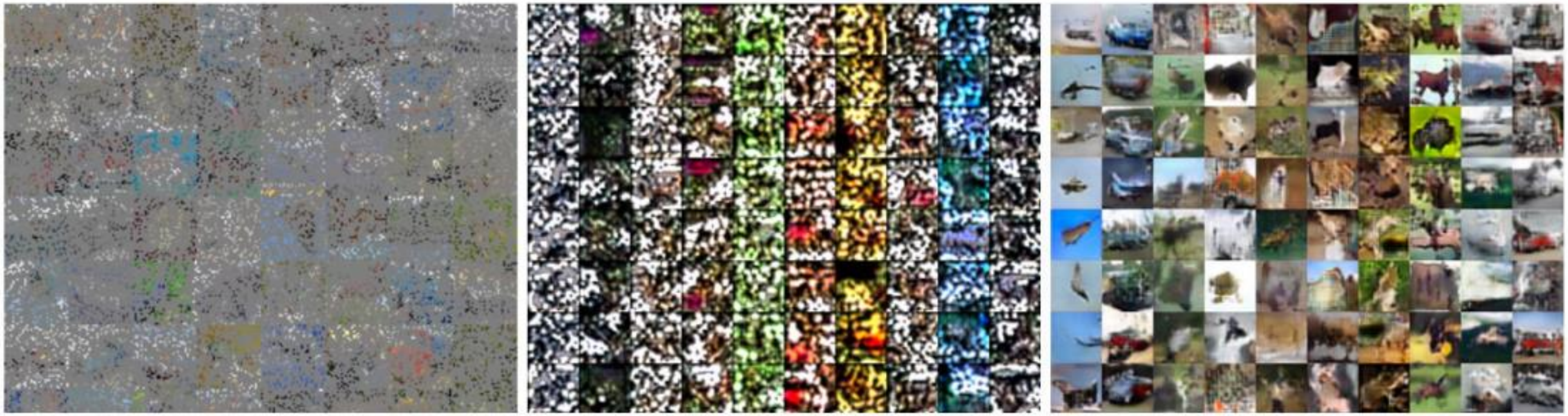
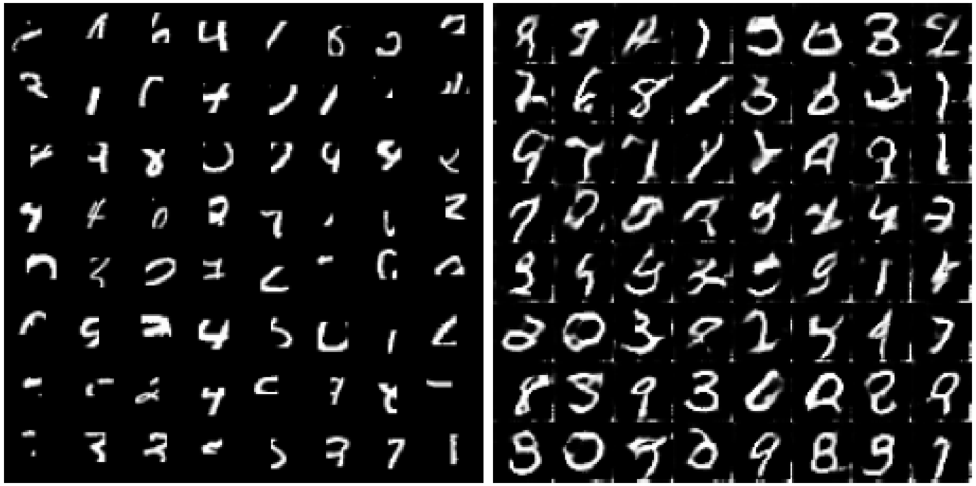
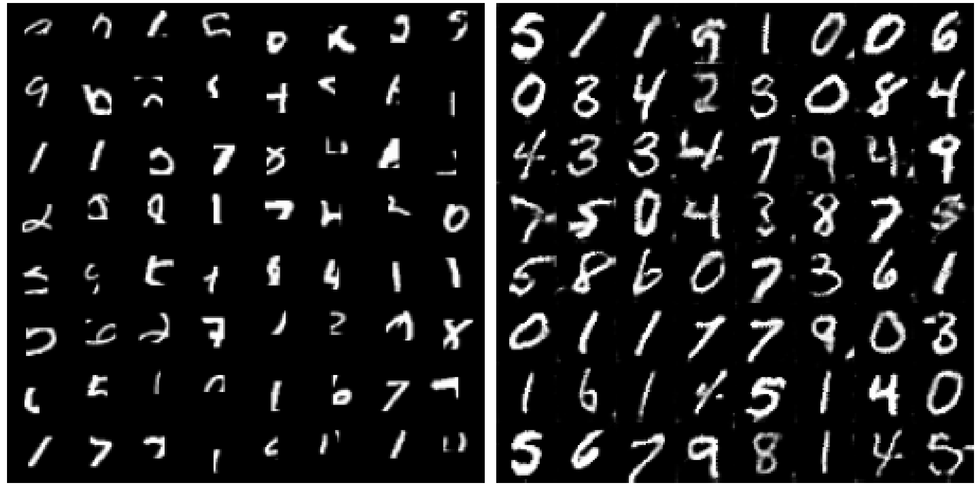


Figure 6: Results with Block-Pixels on CIFAR-10. (left) Samples of lossy measurements. Each pixel is blocked independently with probability $p = 0.8$. Samples produced by (middle) unmeasure-blur baseline, and (right) our model.

Experimental result



(a) (left) Samples of lossy measurements. All except a randomly chosen 14×14 patch is set to zero. (right) Samples produced by the our model.



(b) (left) Samples of lossy measurements. A randomly chosen 14×14 patch is extracted. (right) Samples produced by our model.

Experimental result

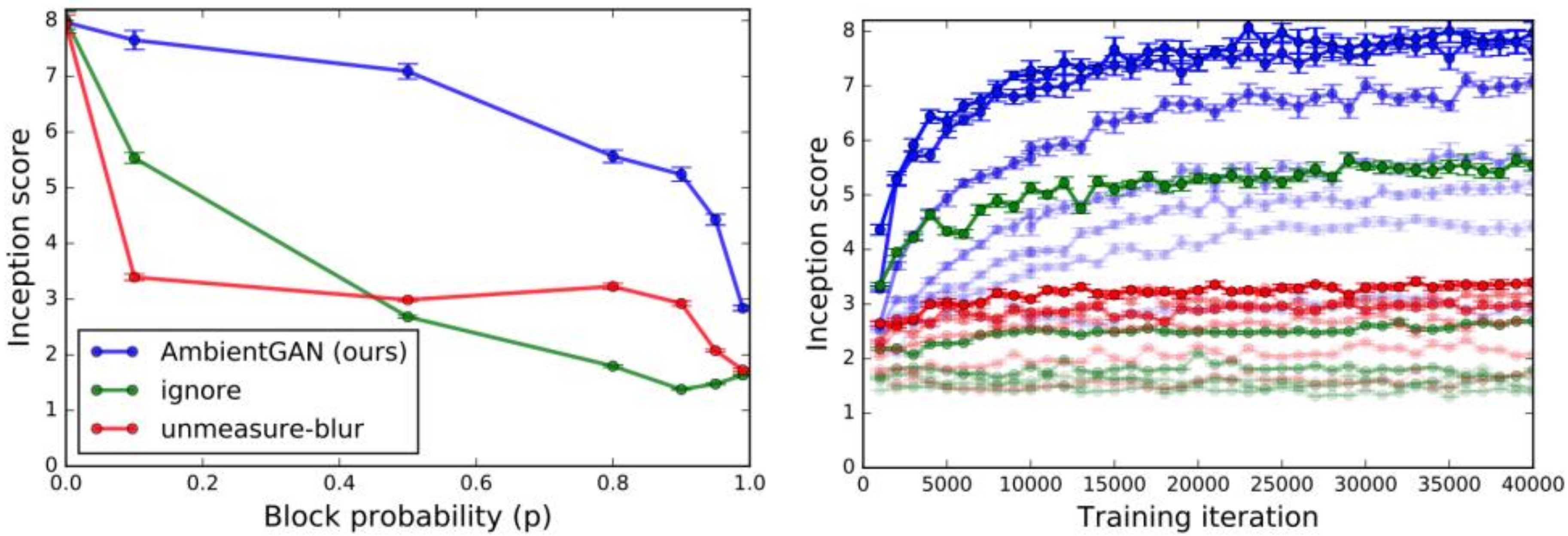


Figure 8: Quantitative results on CIFAR-10 with ACWGANGP, Block-Pixels measurement. (left) Inception score vs blocking probability p. (right) Inception score vs training iteration with darkness proportional to $1 - p$. Vertical bars indicate 95% confidence intervals.

Conclusion

- **It is possible to train the generator without the fully-observed data**
- **Empirically, it is possible to recover the good data distribution even though the measurement process is not clearly known.**
- Possible Applications: OCR, Medical image, etc..

Conclusion

감사합니다