

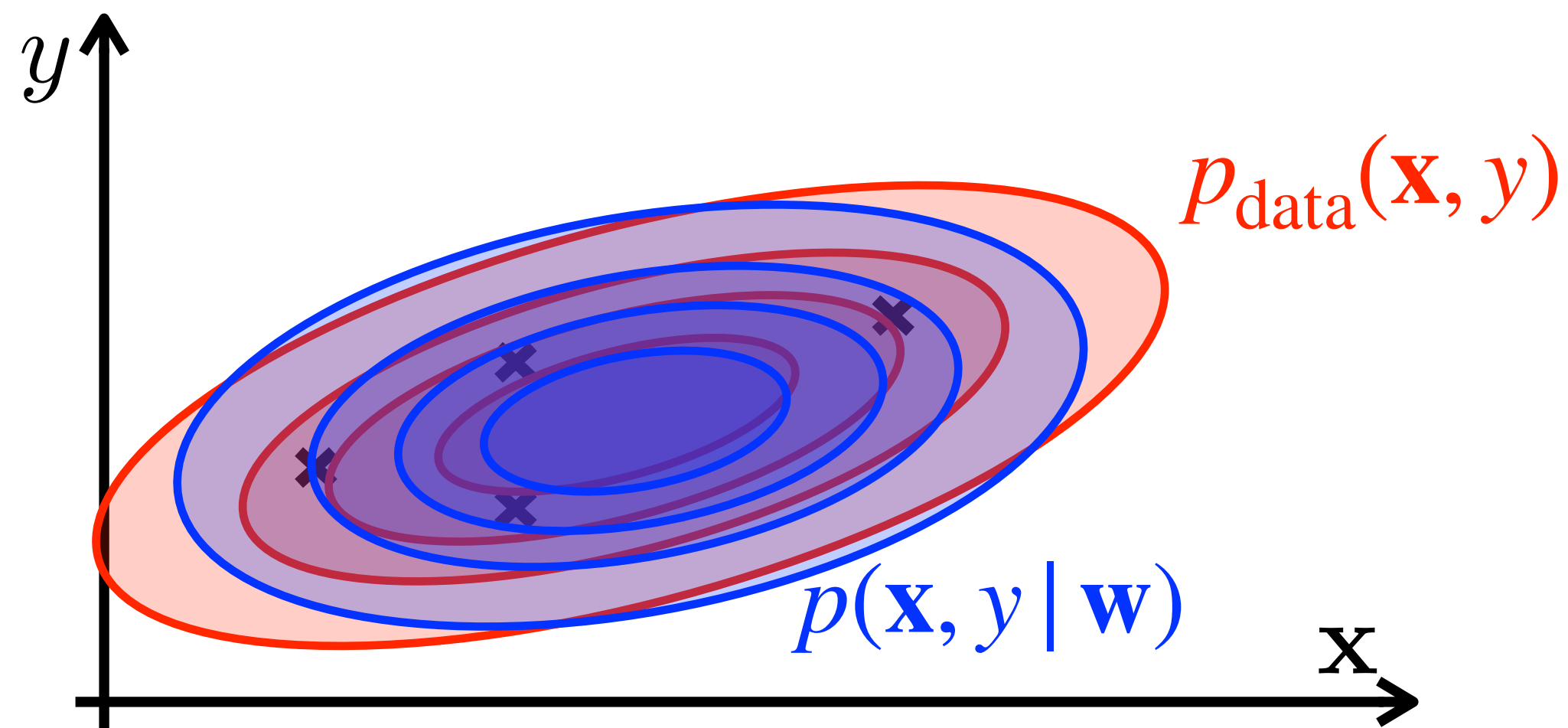
Generative models

VAE, GANS, Diffusion models

Karel Zimmermann

Prerequisites: Learning vs optimization

$$\begin{aligned}\mathbf{w}^\star &= \arg \min_{\mathbf{w}} D_{KL}(\mathbf{p}_{\text{data}}(\mathbf{x}, y) \parallel p(\mathbf{x}, y | \mathbf{w})) = \int_{(\mathbf{x}, y)} \mathbf{p}_{\text{data}}(\mathbf{x}, y) \cdot \log \frac{\mathbf{p}_{\text{data}}(\mathbf{x}, y)}{p(\mathbf{x}, y | \mathbf{w})} \\&= \arg \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x}, y) \sim \mathbf{p}_{\text{data}}} \left[\log \frac{\mathbf{p}_{\text{data}}(\mathbf{x}, y)}{p(\mathbf{x}, y | \mathbf{w})} \right] = \arg \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x}, y) \sim \mathbf{p}_{\text{data}}} \left[-\log p(\mathbf{x}, y | \mathbf{w}) \right] \\&= \arg \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x}, y) \sim \mathbf{p}_{\text{data}}} \left[\log \mathbf{p}_{\text{data}}(\mathbf{x}, y) - \log p(y | \mathbf{x}, \mathbf{w}) p(\mathbf{x}) \right] \\&= \arg \min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x}, y) \sim \mathbf{p}_{\text{data}}} \left[-\log p(y | \mathbf{x}, \mathbf{w}) \right] \approx \arg \min_{\mathbf{w}} \frac{1}{N} \sum_{(\mathbf{x}_i, y_i) \sim \mathbf{p}_{\text{data}}(\mathbf{x}, y)} \left[-\log p(y_i | \mathbf{x}_i, \mathbf{w}) \right]\end{aligned}$$

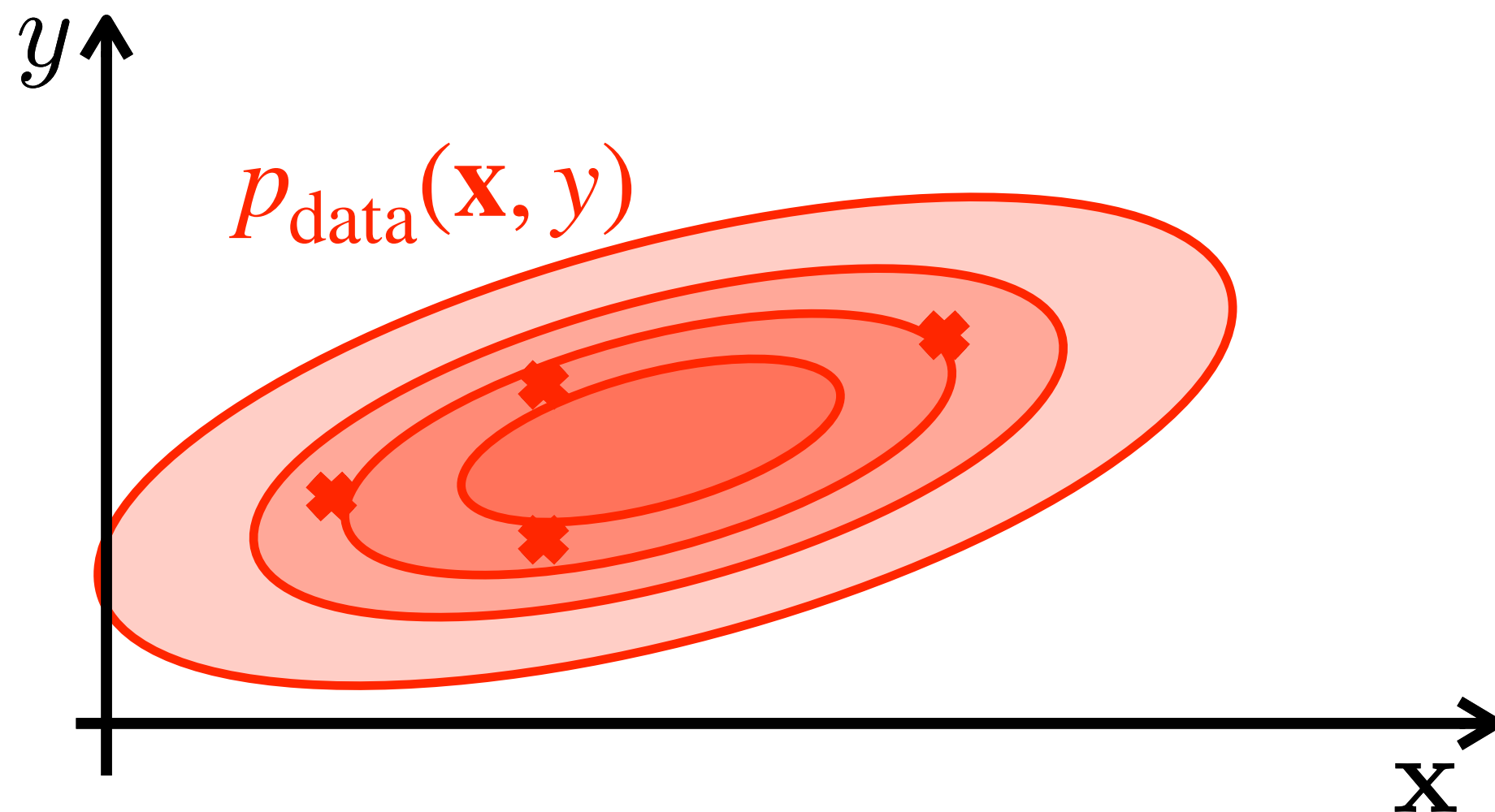
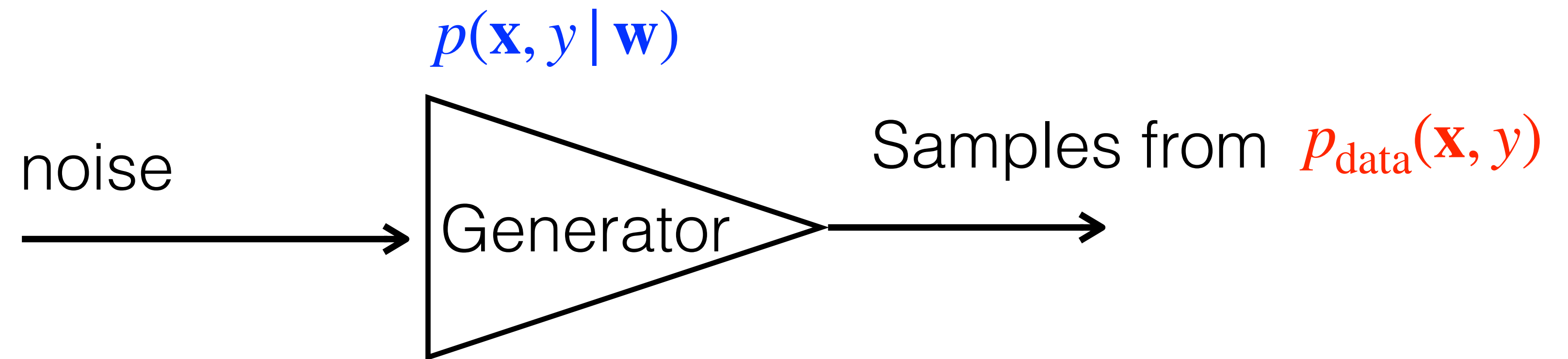


So-far we always modeled $p(y_i | \mathbf{x}_i, \mathbf{w})$
How comes?

What else can be modeled?

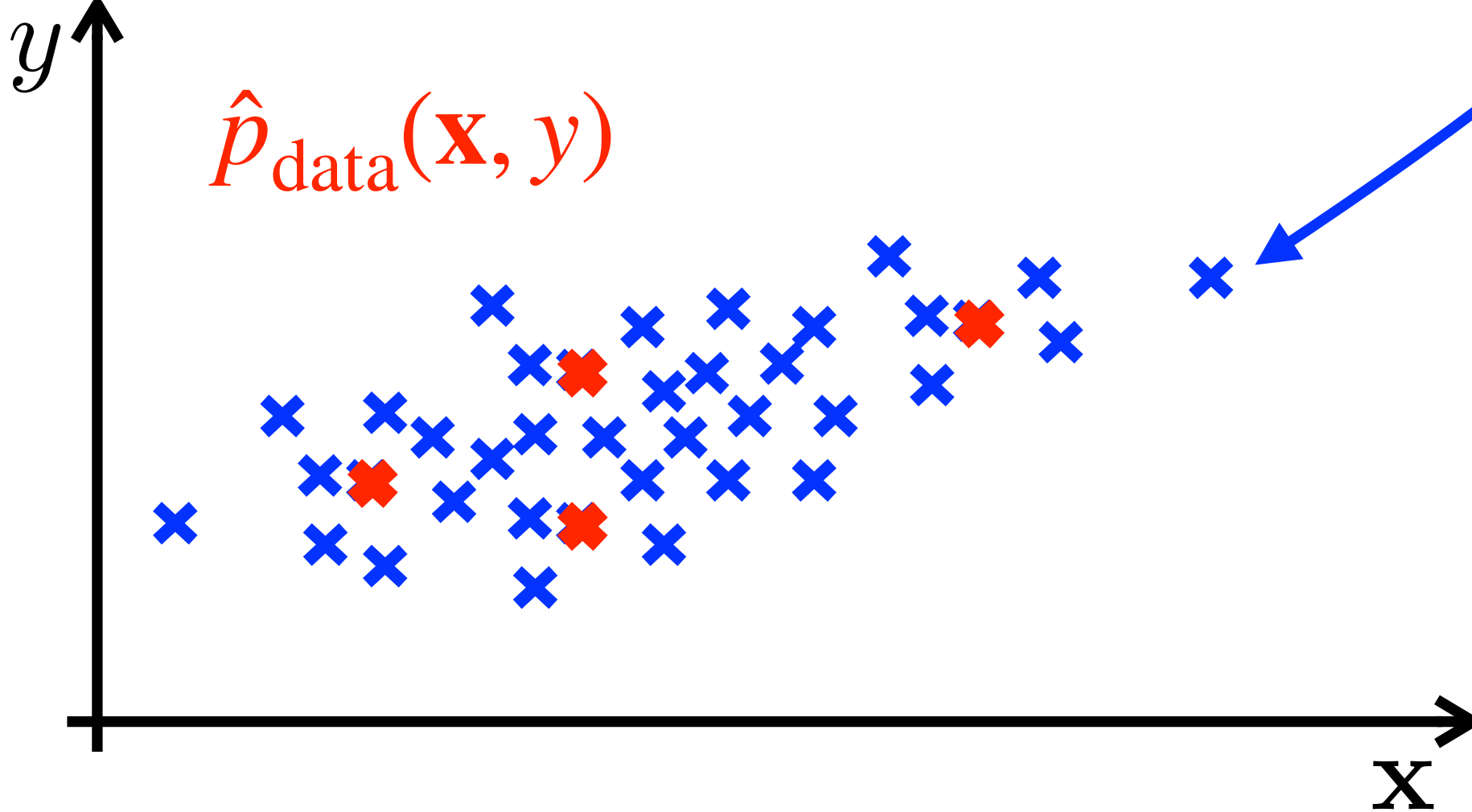
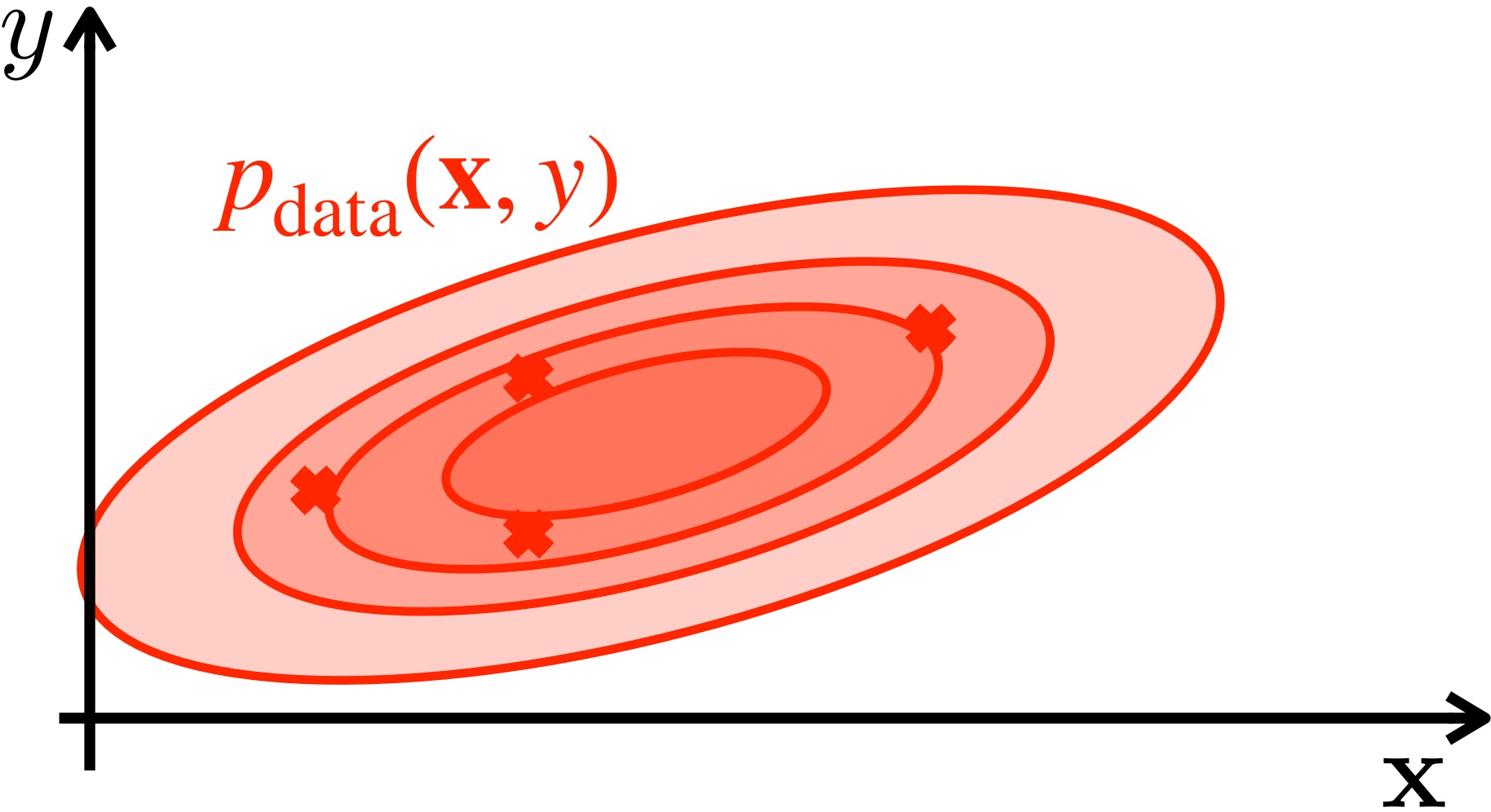
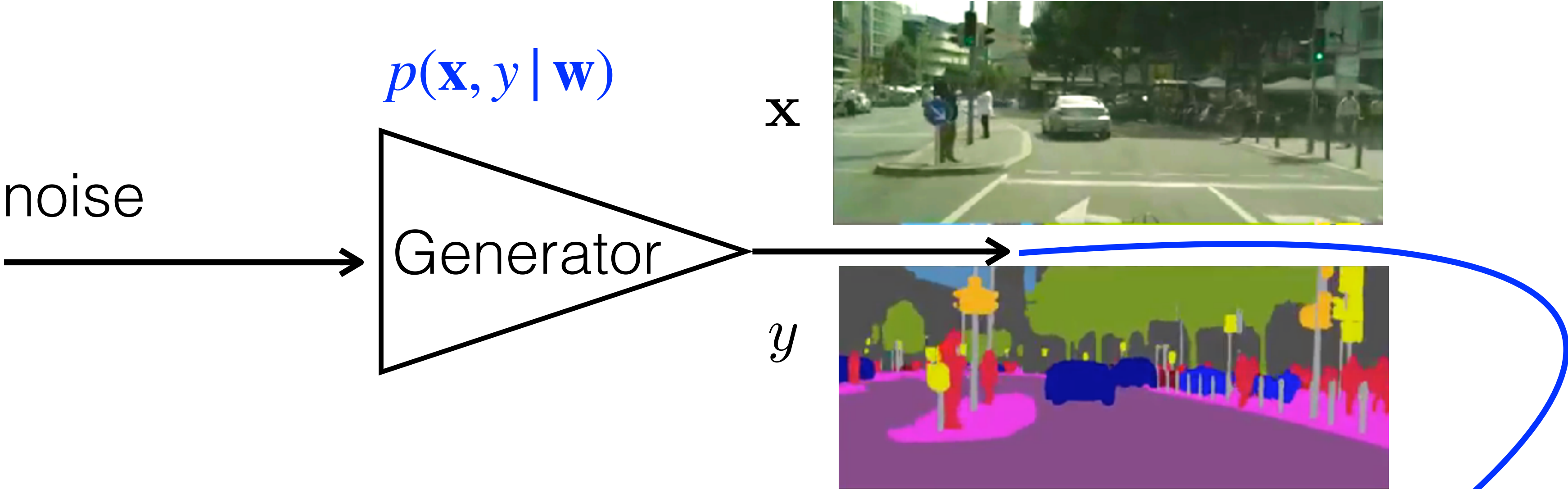
$p(\mathbf{x}, y | \mathbf{w})$ $p(\mathbf{x} | y, \mathbf{w})$ $p(\mathbf{x} | \mathbf{w})$

Dataset augmentation



Dataset augmentation

Samples from $p_{\text{data}}(\mathbf{x}, y)$



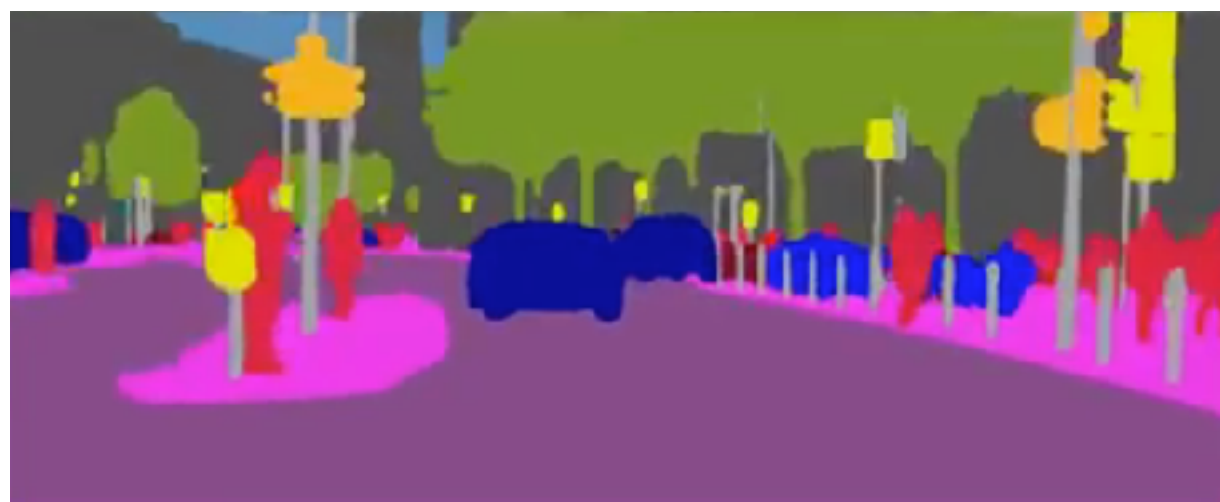
Dataset augmentation

Samples from $\hat{p}_{\text{data}}(\mathbf{x} | y)$

annotation

photo

y



$p(\mathbf{x} | y, \mathbf{w})$

Generator

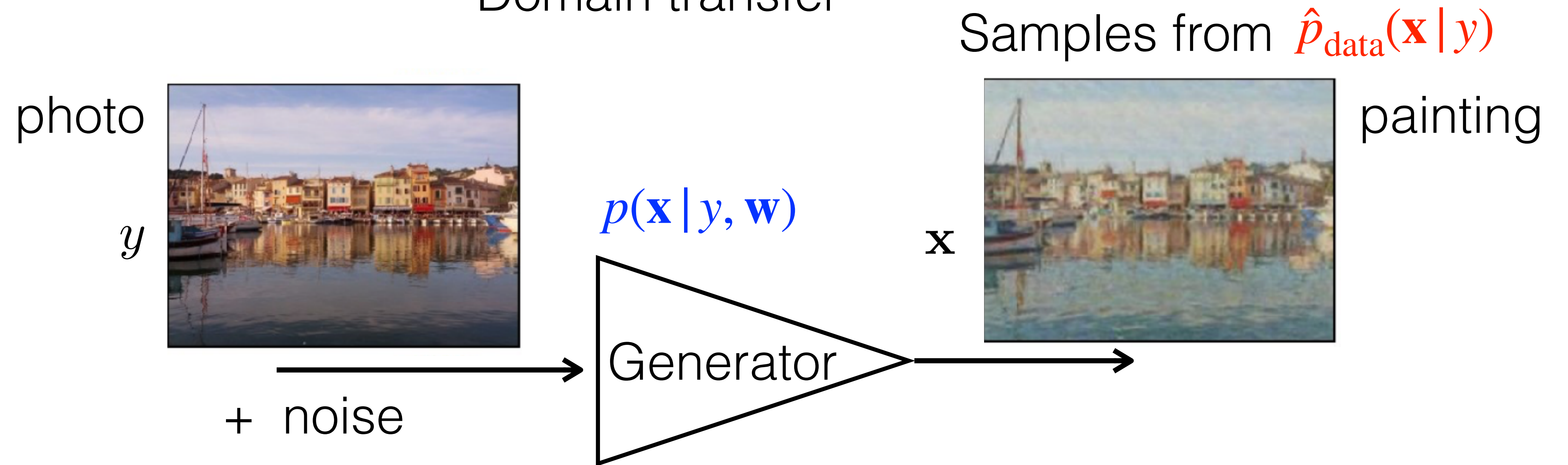
$\mathbf{x}$



+ noise

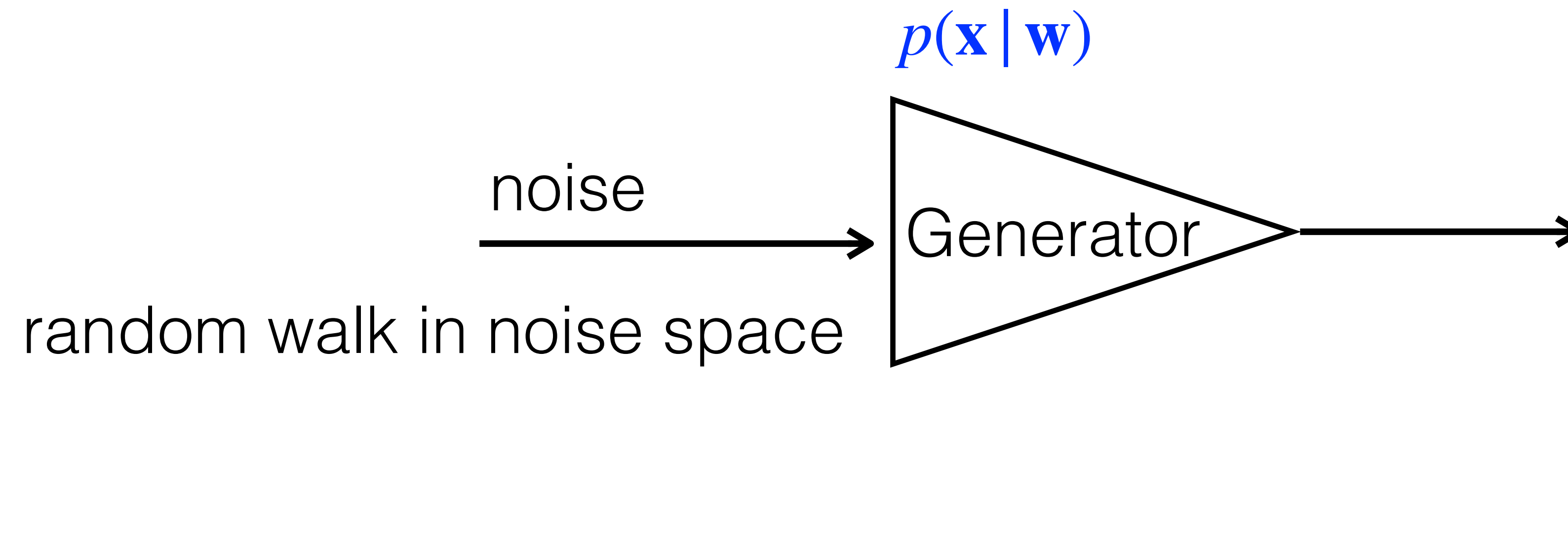
Gives better control over generated data

Domain transfer



Classification dataset augmentation

Samples from $\hat{p}_{\text{data}}(\mathbf{x})$



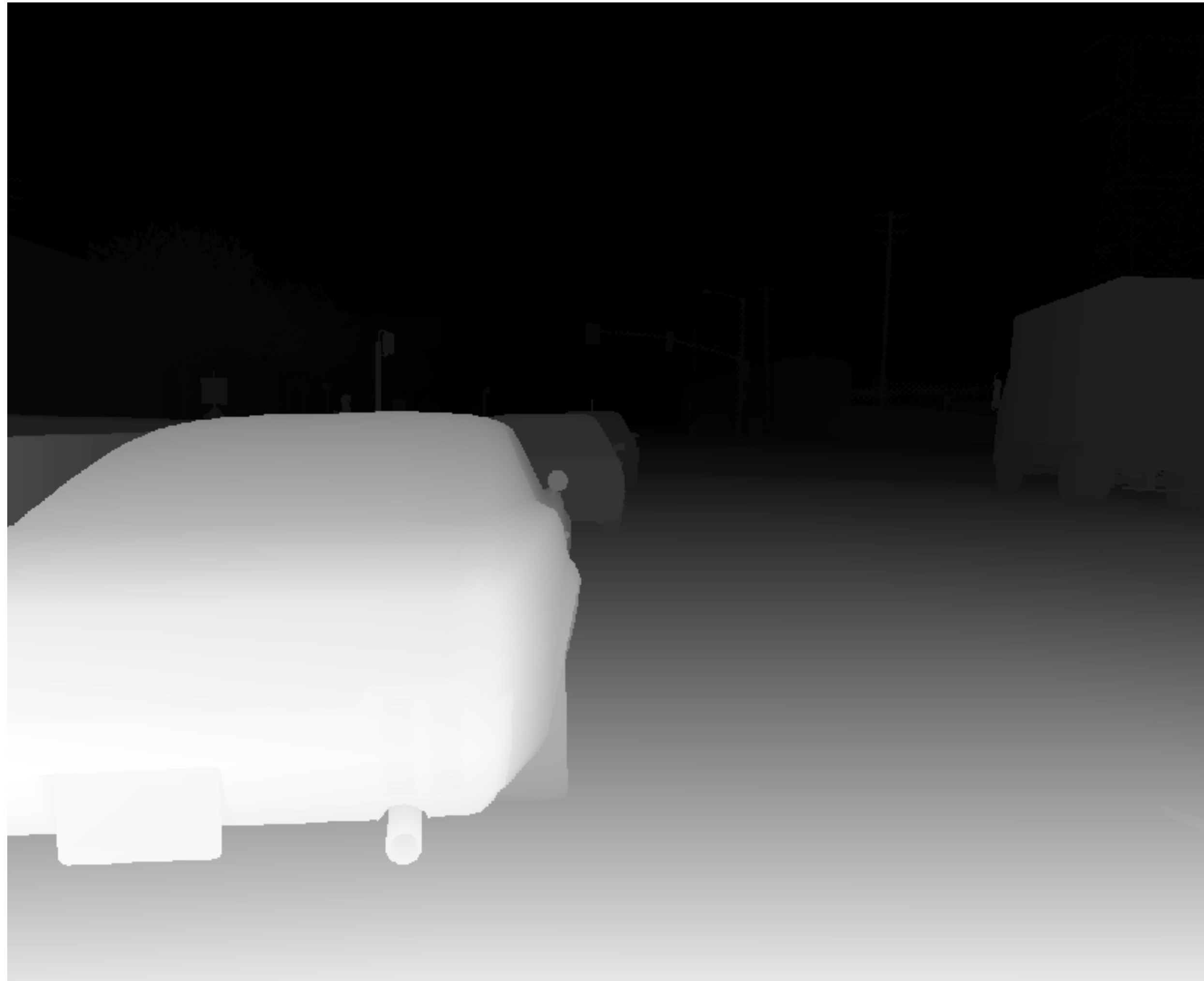
How can I do it?

- Dataset augmentation
- Realistic simulator
- Generative networks

RGB images



Depth images



Stencil layer



Stencil layer - **cars**



Stencil layer - **humans**



Stencil layer - **trees**



Stencil layer - **sky**



Stencil layer - **artificial light**



Stencil layer - **artificial light**



Other annotations for objects (e.g. cars, humans)



Other annotations for objects (e.g. cars, humans)



What can be controlled (night/day)



What can be controlled (night/day)



What can be controlled (weather): ExtraSunny



What can be controlled (weather): Clear



What can be controlled (weather): Foggy



```
handlePipeInput called  
server connected:False  
connection:System.Net.Sockets.Socket
```


What can be controlled (weather): OverCast



What can be controlled (weather): Raining



handlePipeInput called
N pressed, going to take
screenshots
server connected:False
connection:System.Net.Sockets.Socket

What can be controlled (weather): ThunderStorm



What can be controlled (weather): Clearing



What can be controlled (weather): SnowLight



What can be controlled (mods): Tsunami



<https://cs.gta5-mods.com/misc/tsunami-mod>

What can be controlled (mods): Plane crashing mods

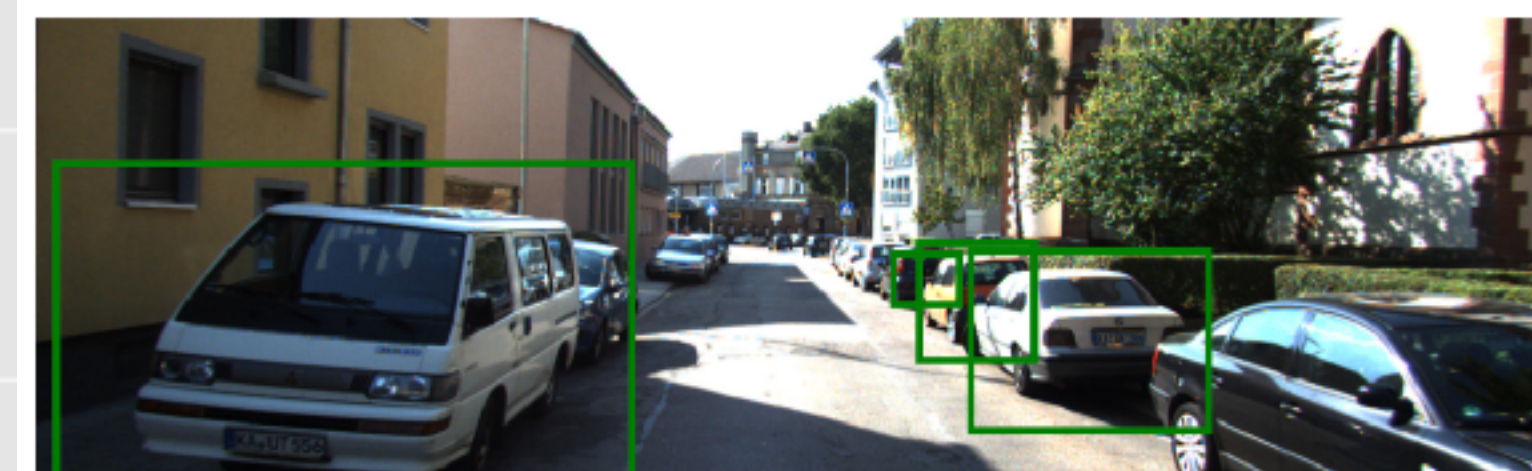
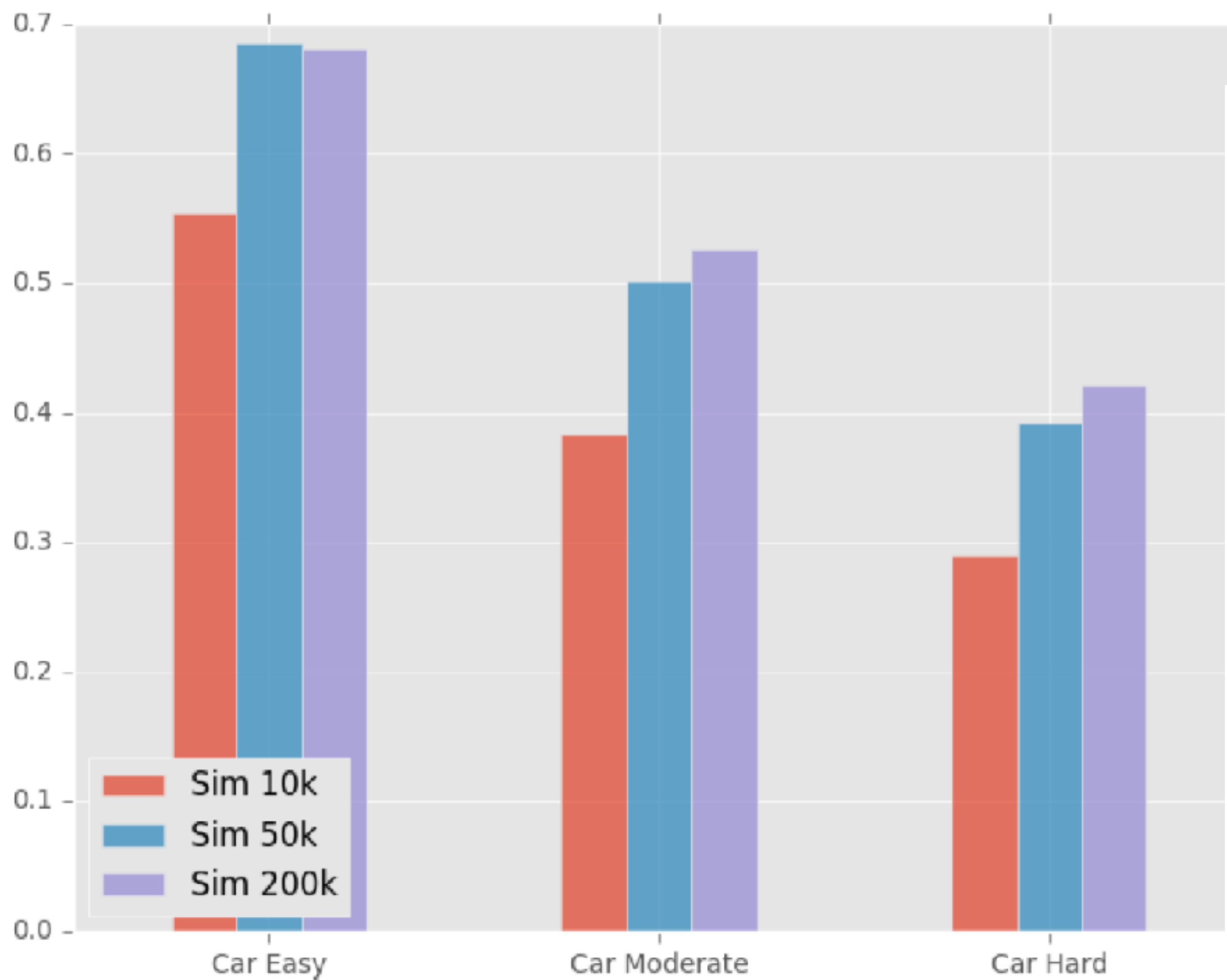


<https://cs.gta5-mods.com/scripts/planes-hails>

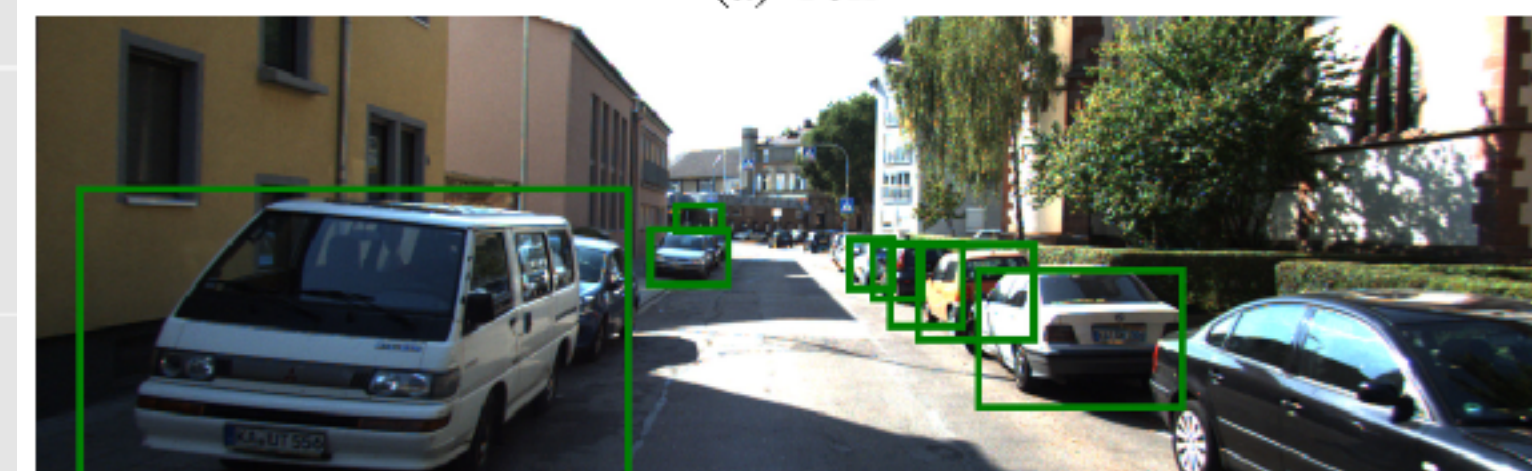
Driving in the matrix [Roberson ICRA 2017]

<https://arxiv.org/abs/1610.01983>

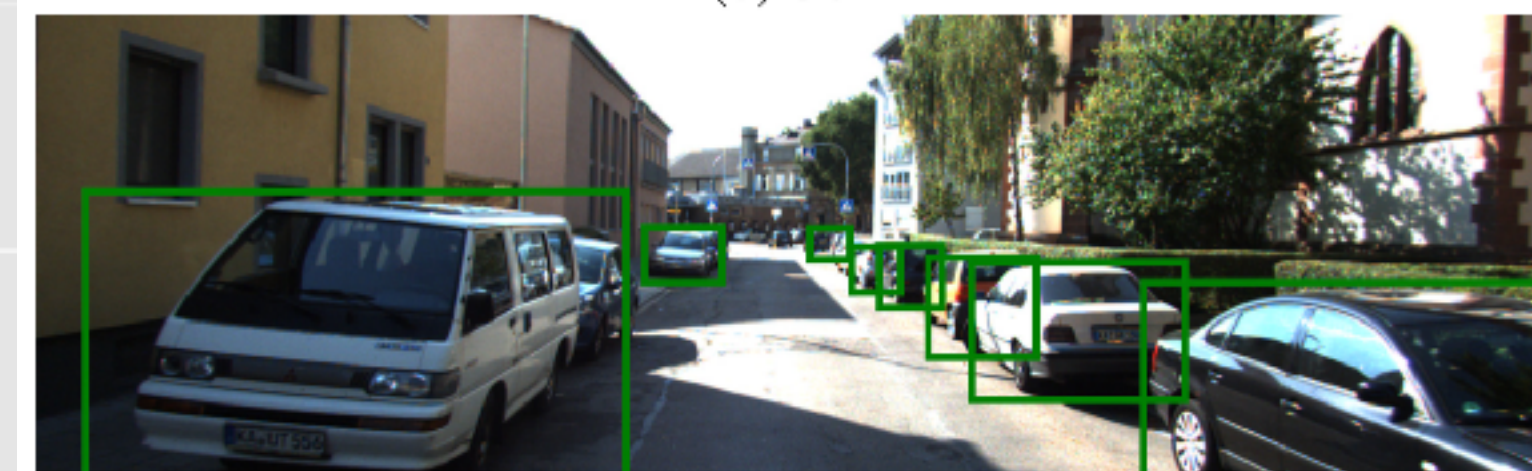
- Reverse engineering of GTA 5 (RAGE engine)



(a) 10k



(b) 50k

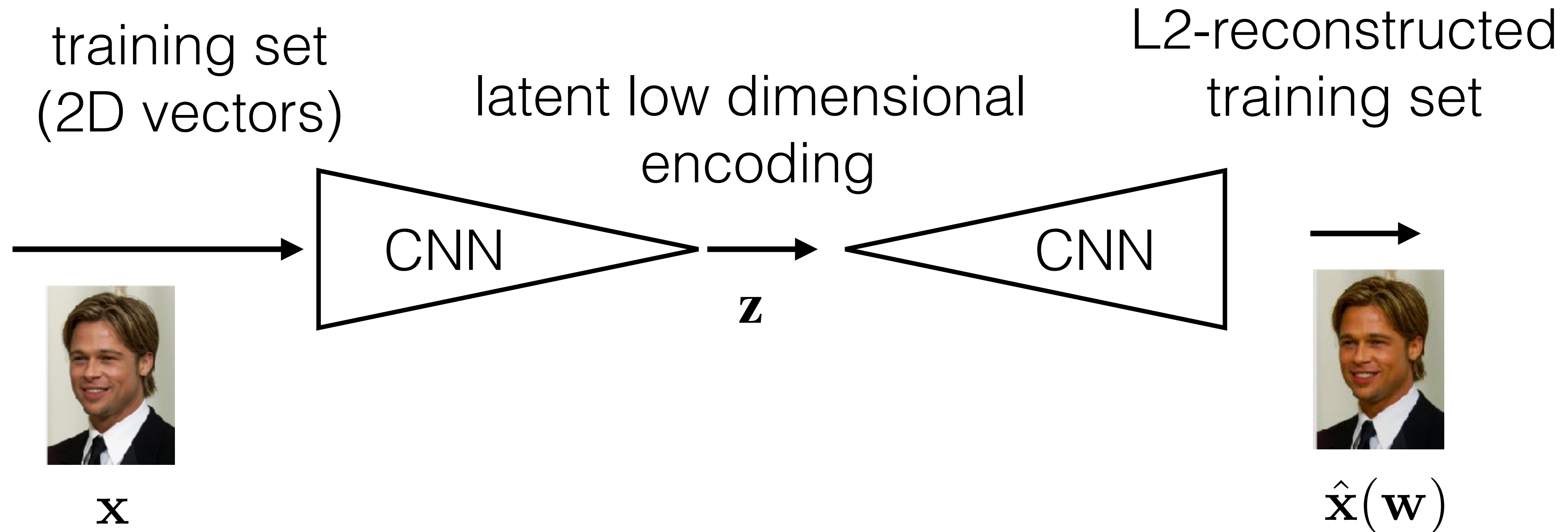


(c) 200k

How can I do it?

- Dataset augmentation
- Realistic simulator
- Generative networks

Generative models



- Learning the self-reconstruction with L2 reconstruction loss

$$\arg \min_{\mathbf{w}} \|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{w})\|_2^2$$

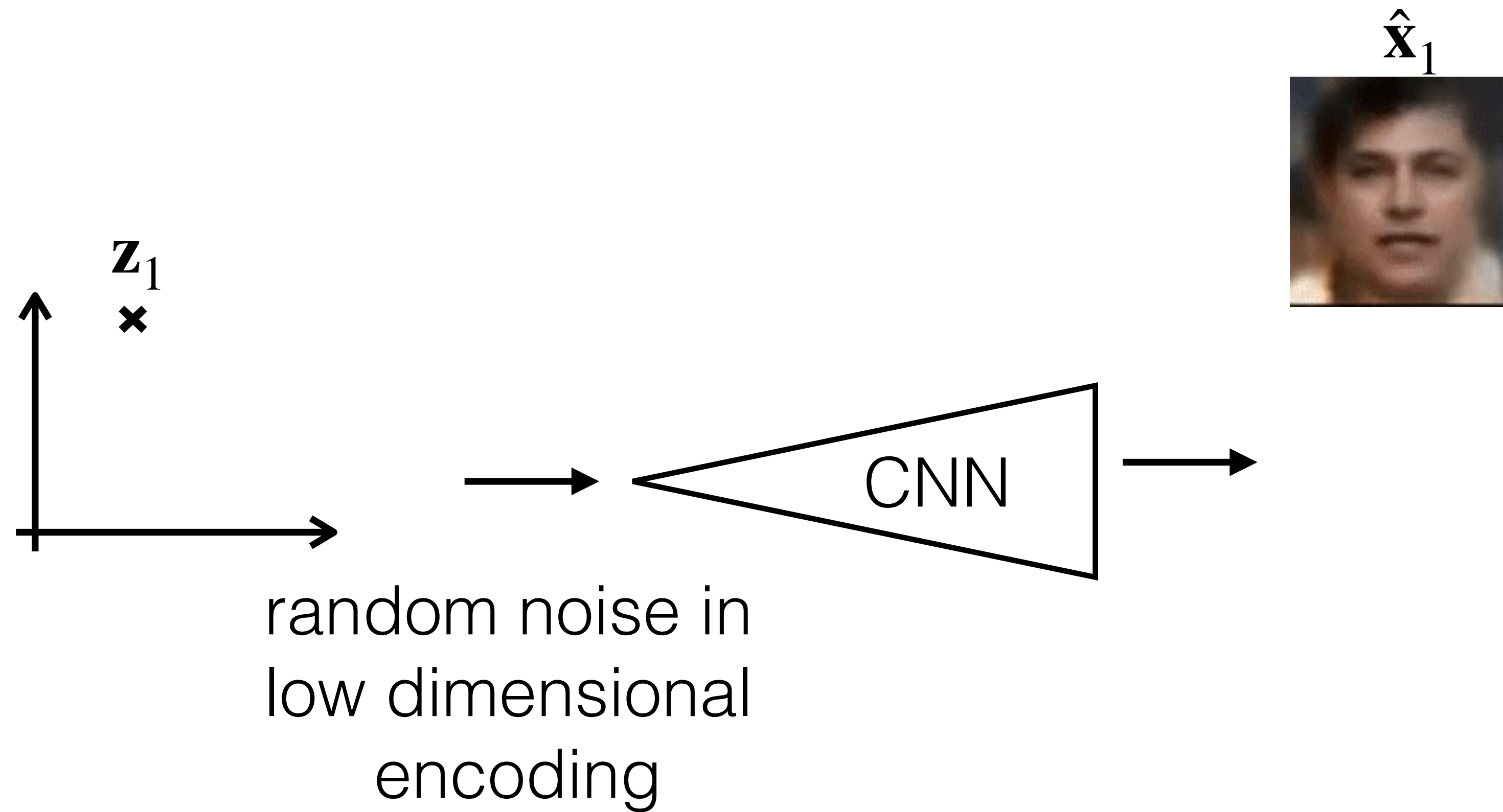
If CNN is pure linear function then:

- closed-form solution exists
- method is called PCA

If $\mathbf{z}$ is pushed towards gaussian distribution:

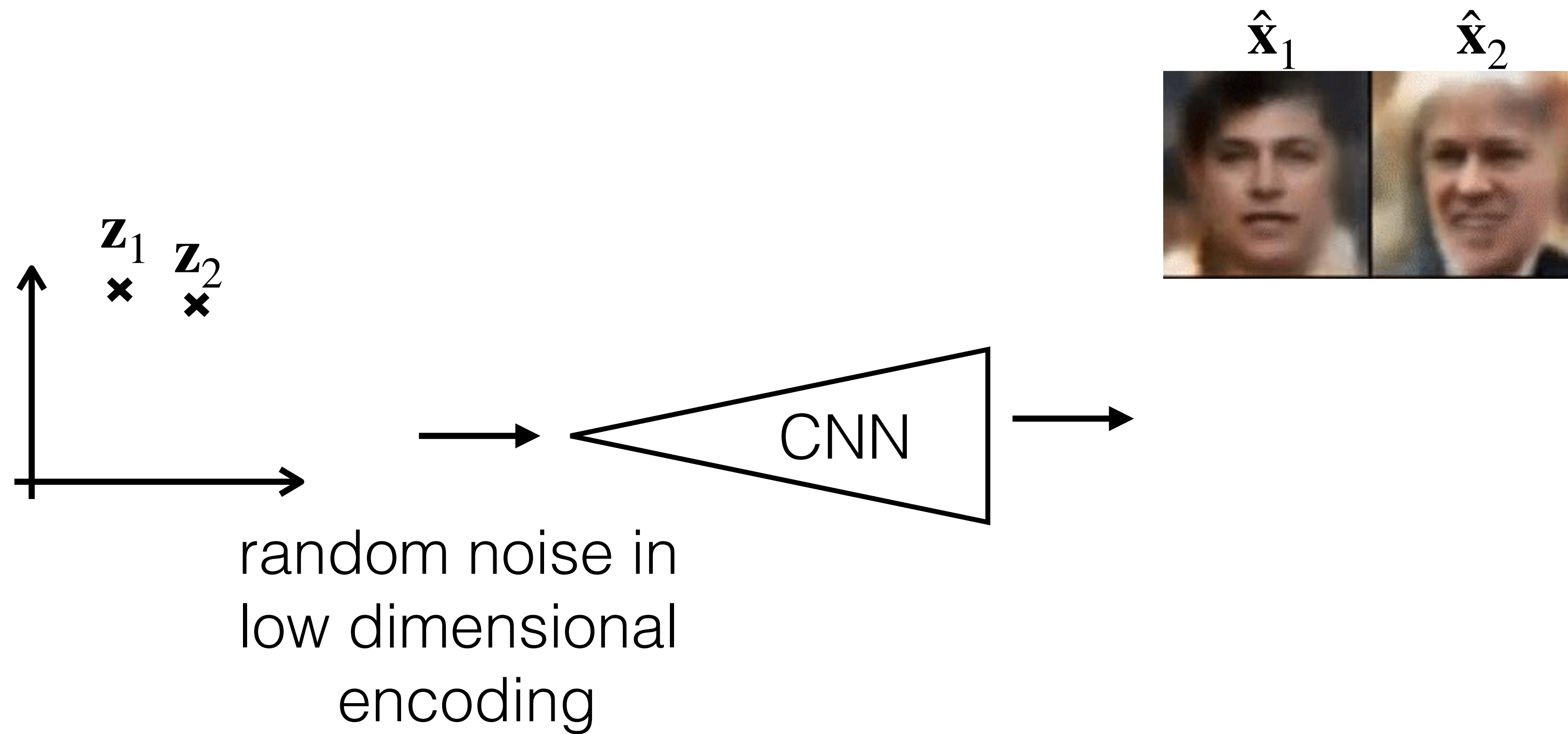
- method is referred as variational encoders

Generative models



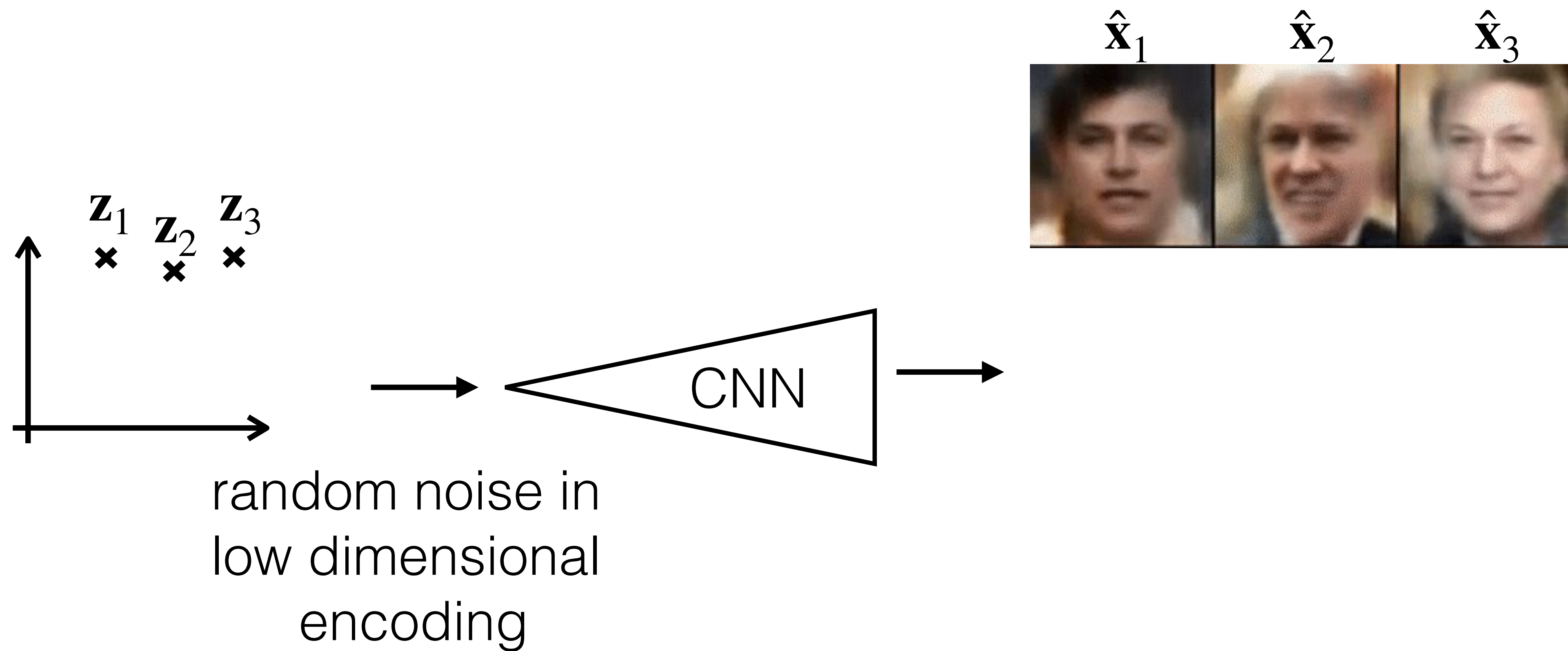
- New samples generated from random vectors in low-dimensional encoding.

Generative models



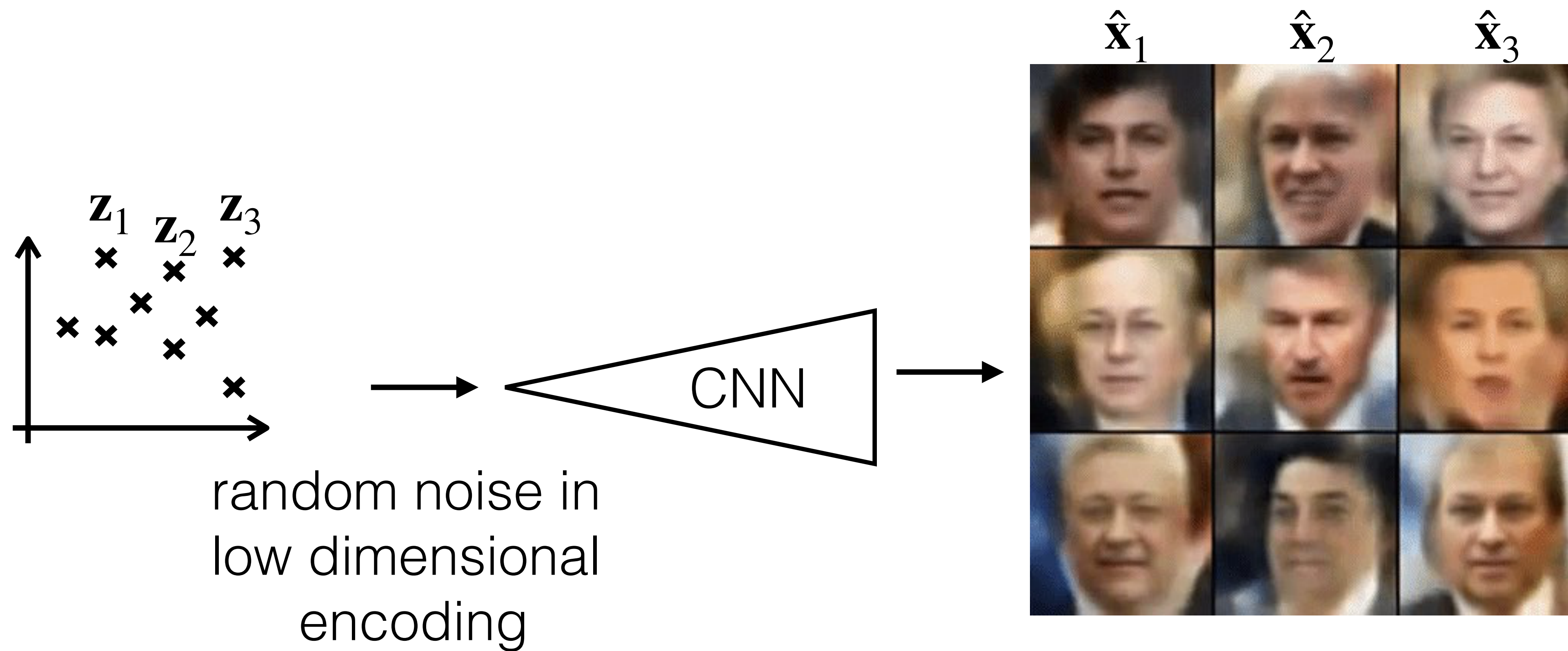
- New samples generated from random vectors in low-dimensional encoding.

Generative models



- New samples generated from random vectors in low-dimensional encoding.

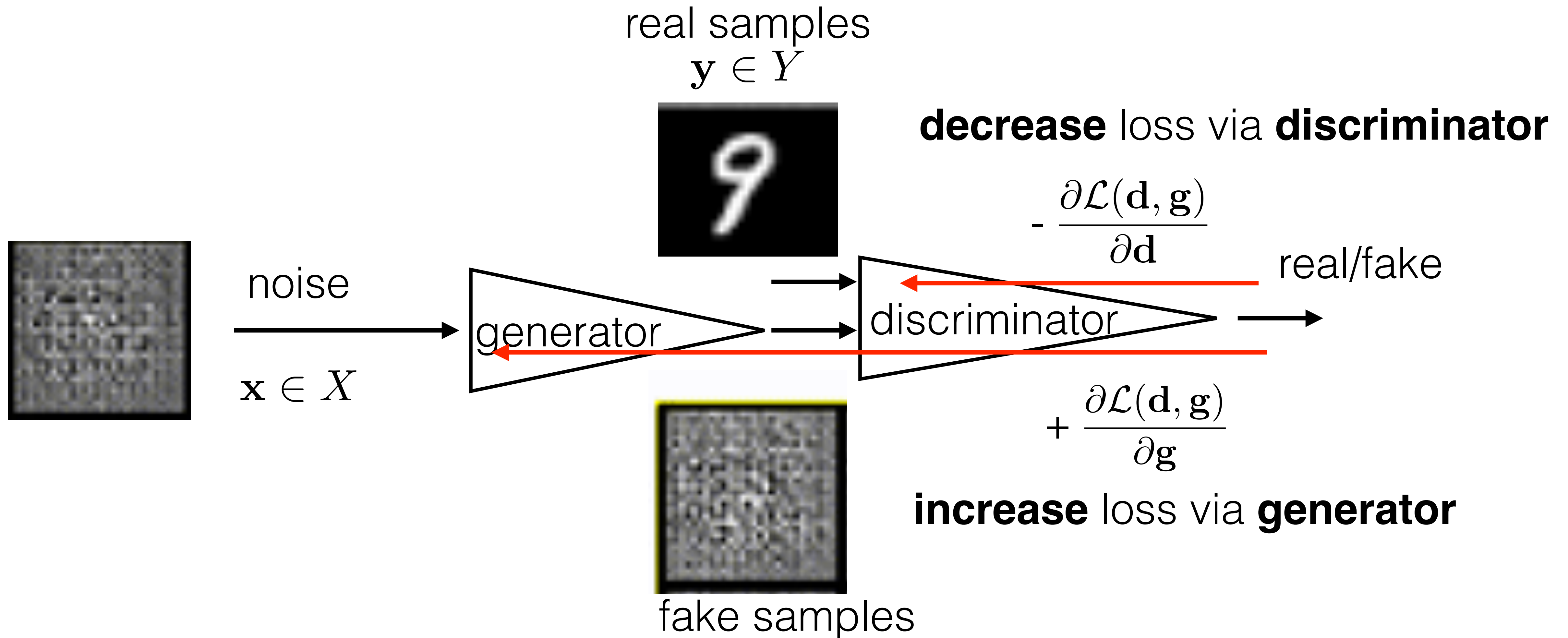
Generative models



- New samples generated from random vectors in low-dimensional encoding.

Generative Adversarial Nets [Goodfellow NIPS 2014]

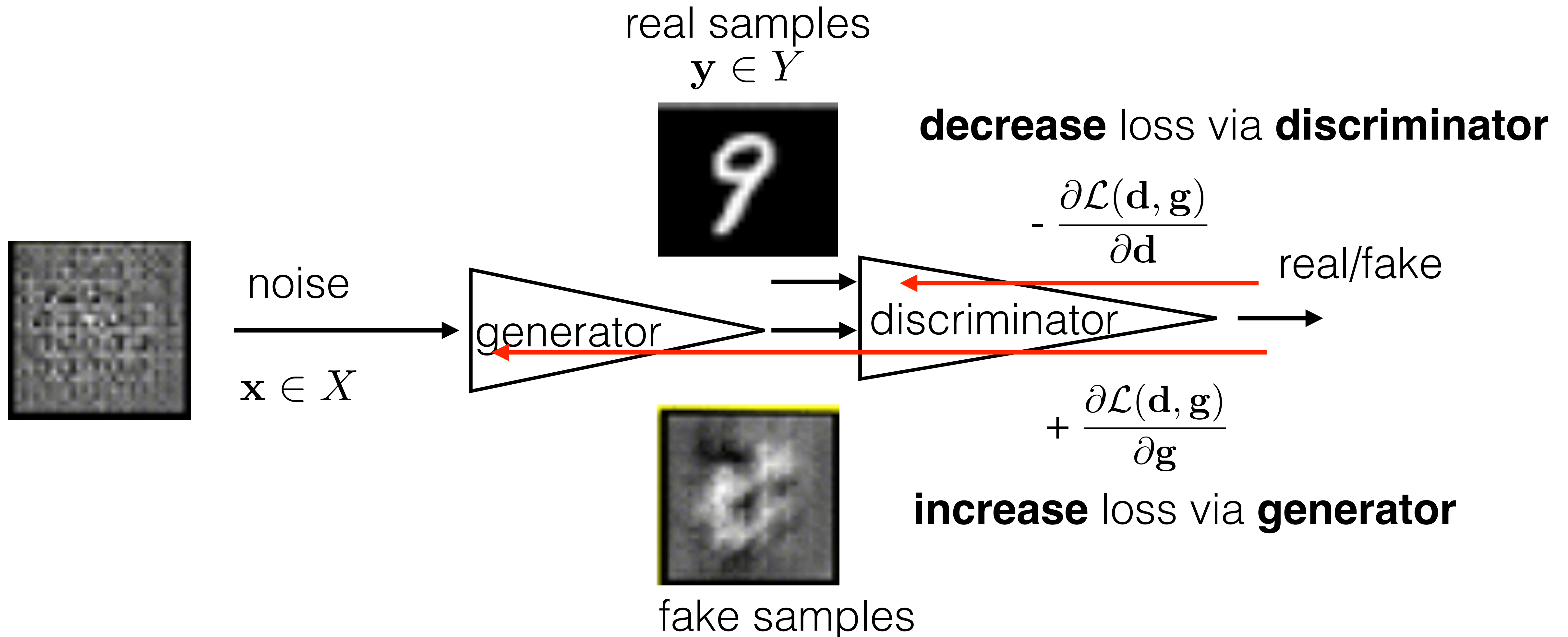
<https://arxiv.org/abs/1406.2661>



classification loss:
$$\mathcal{L}(\mathbf{d}, \mathbf{g}) = \sum_{\mathbf{x} \in X} -\log(\mathbf{d}(\mathbf{g}(\mathbf{x}))) + \sum_{\mathbf{y} \in Y} -\log(1 - \mathbf{d}(\mathbf{y}))$$

Generative Adversarial Nets [Goodfellow NIPS 2014]

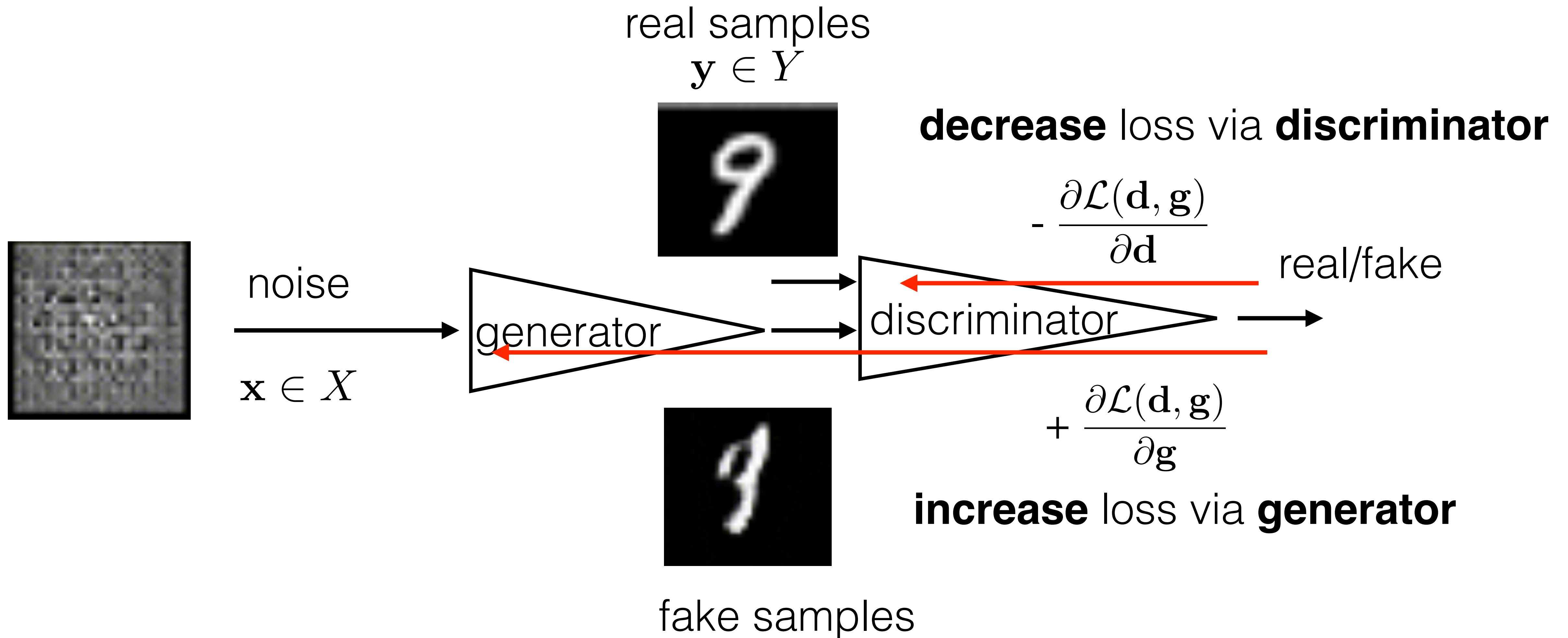
<https://arxiv.org/abs/1406.2661>



classification loss:
$$\mathcal{L}(\mathbf{d}, \mathbf{g}) = \sum_{\mathbf{x} \in X} -\log(\mathbf{d}(\mathbf{g}(\mathbf{x}))) + \sum_{\mathbf{y} \in Y} -\log(1 - \mathbf{d}(\mathbf{y}))$$

Generative Adversarial Nets [Goodfellow NIPS 2014]

<https://arxiv.org/abs/1406.2661>



classification loss:
$$\mathcal{L}(\mathbf{d}, \mathbf{g}) = \sum_{\mathbf{x} \in X} -\log(\mathbf{d}(\mathbf{g}(\mathbf{x}))) + \sum_{\mathbf{y} \in Y} -\log(1 - \mathbf{d}(\mathbf{y}))$$

Generative Adversarial Nets [Goodfellow NIPS 2014]

<https://arxiv.org/abs/1406.2661>

real samples
 $\mathbf{y} \in Y$

$$(\mathbf{d}^*, \mathbf{g}^*) = \arg \min_{\mathbf{d}} \arg \max_{\mathbf{g}} \mathcal{L}(\mathbf{d}, \mathbf{g})$$

decrease loss via **discriminator**



noise

$\mathbf{x} \in X$

generator



fake samples

discriminator

real/fake

$$- \frac{\partial \mathcal{L}(\mathbf{d}, \mathbf{g})}{\partial \mathbf{d}}$$

$$+ \frac{\partial \mathcal{L}(\mathbf{d}, \mathbf{g})}{\partial \mathbf{g}}$$

increase loss via **generator**

classification loss:
$$\mathcal{L}(\mathbf{d}, \mathbf{g}) = \sum_{\mathbf{x} \in X} -\log(\mathbf{d}(\mathbf{g}(\mathbf{x}))) + \sum_{\mathbf{y} \in Y} -\log(1 - \mathbf{d}(\mathbf{y}))$$

Generative Adversarial Nets [Goodfellow NIPS 2014]

<https://arxiv.org/abs/1406.2661>

$$(\mathbf{d}^*, \mathbf{g}^*) = \arg \min_{\mathbf{d}} \arg \max_{\mathbf{g}} \mathcal{L}(\mathbf{d}, \mathbf{g})$$

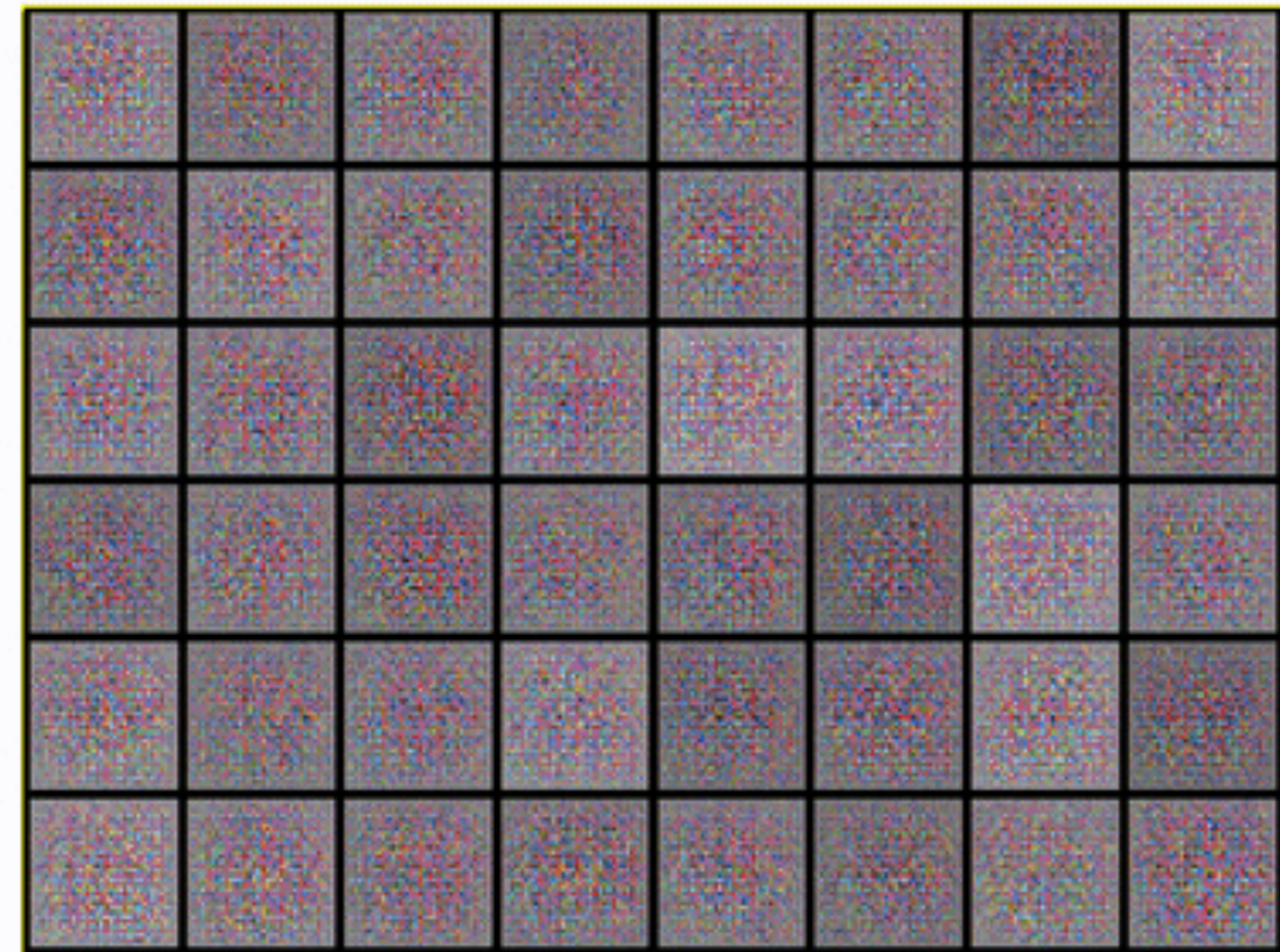
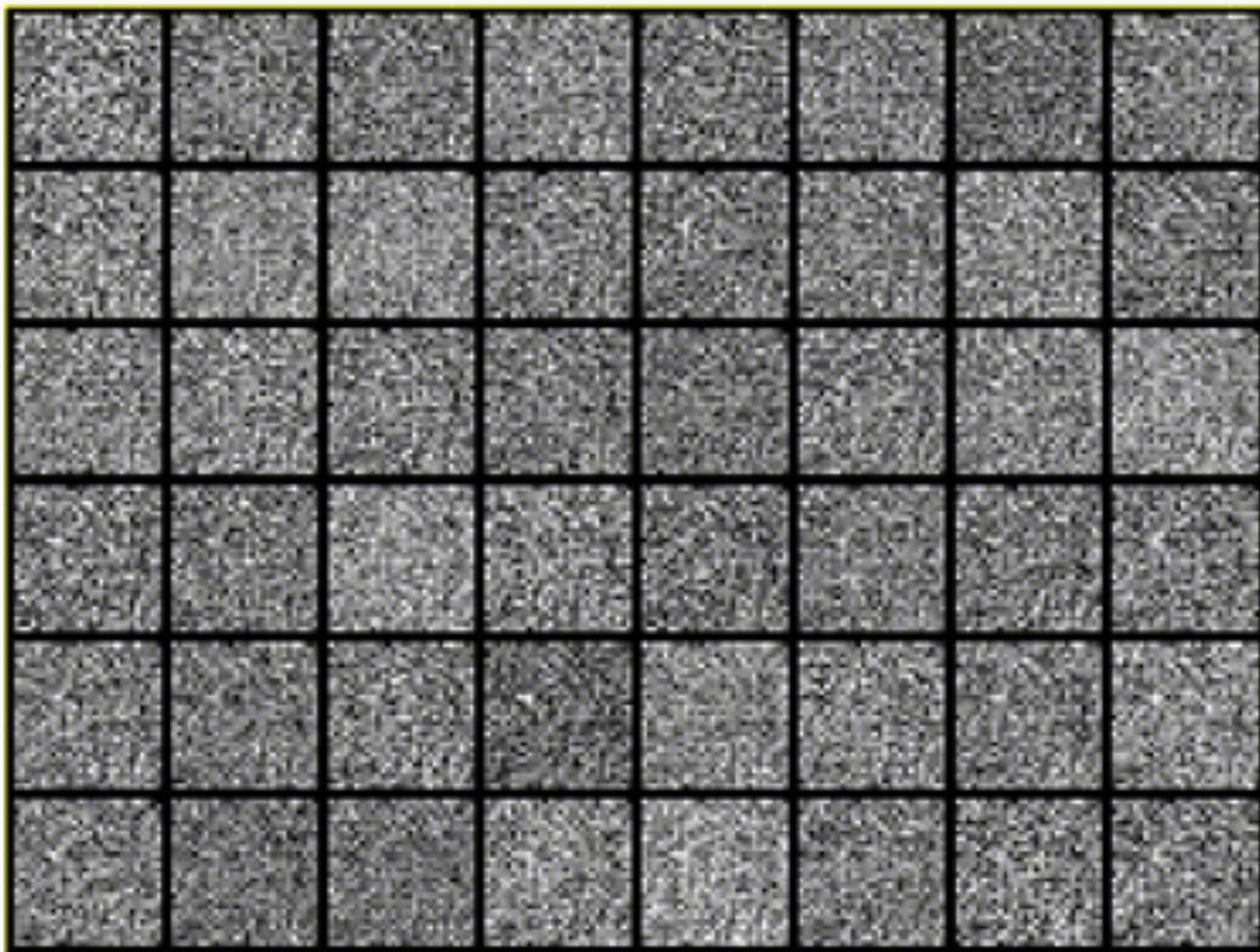
- Equilibrium in **saddle** point **implies** that generator generates samples from the **real distribution** (asymptotically consistent in contrast to VAE)

Generative Adversarial Nets [Goodfellow NIPS 2014]

<https://arxiv.org/abs/1406.2661>

$$(\mathbf{d}^*, \mathbf{g}^*) = \arg \min_{\mathbf{d}} \arg \max_{\mathbf{g}} \mathcal{L}(\mathbf{d}, \mathbf{g})$$

- Equilibrium in **saddle** point **implies** that generator generates samples from the **real distribution** (asymptotically consistent in contrast to VAE)



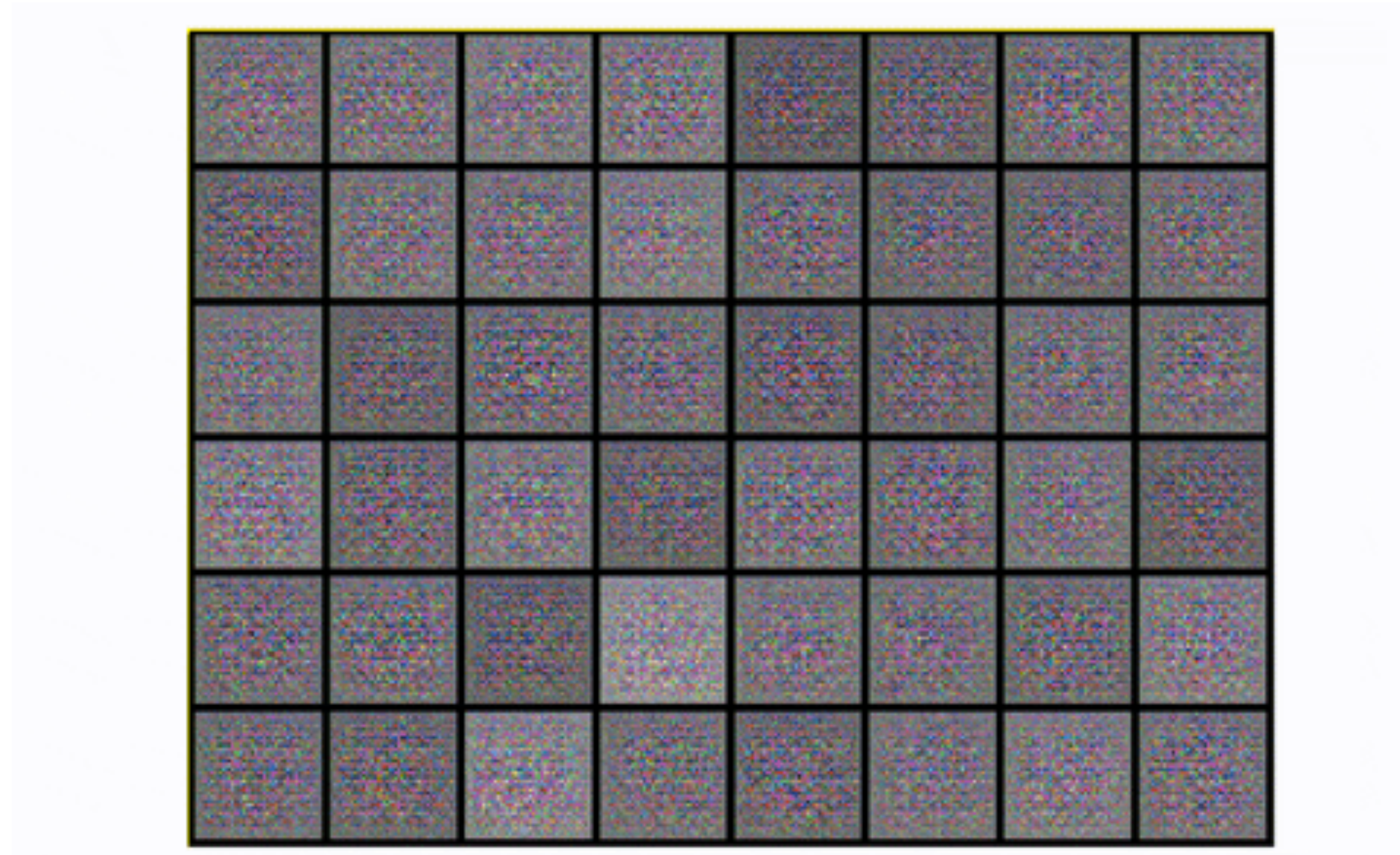
<https://theaisummer.com/gan-computer-vision/>

Generative Adversarial Nets [Goodfellow NIPS 2014]

<https://arxiv.org/abs/1406.2661>

$$(\mathbf{d}^*, \mathbf{g}^*) = \arg \min_{\mathbf{d}} \arg \max_{\mathbf{g}} \mathcal{L}(\mathbf{d}, \mathbf{g})$$

- The learning is generally unstable and suffers from mode collapse



<https://theaisummer.com/gan-computer-vision/>

Domain transfer

Cycle-GAN [Zhu ICCV 2017]

<https://arxiv.org/abs/1703.10593>



real Monet's painting

real Y-domain

samples $y \in Y$

real/fake

$$\mathcal{L}(\mathbf{d}_y, \mathbf{g}_x)$$

$\mathbf{g}_x(\mathbf{x})$

$\mathbf{d}_y(\mathbf{y})$

real X-domain
samples $\mathbf{x} \in X$

fake Y-domain
samples $\hat{\mathbf{y}}$



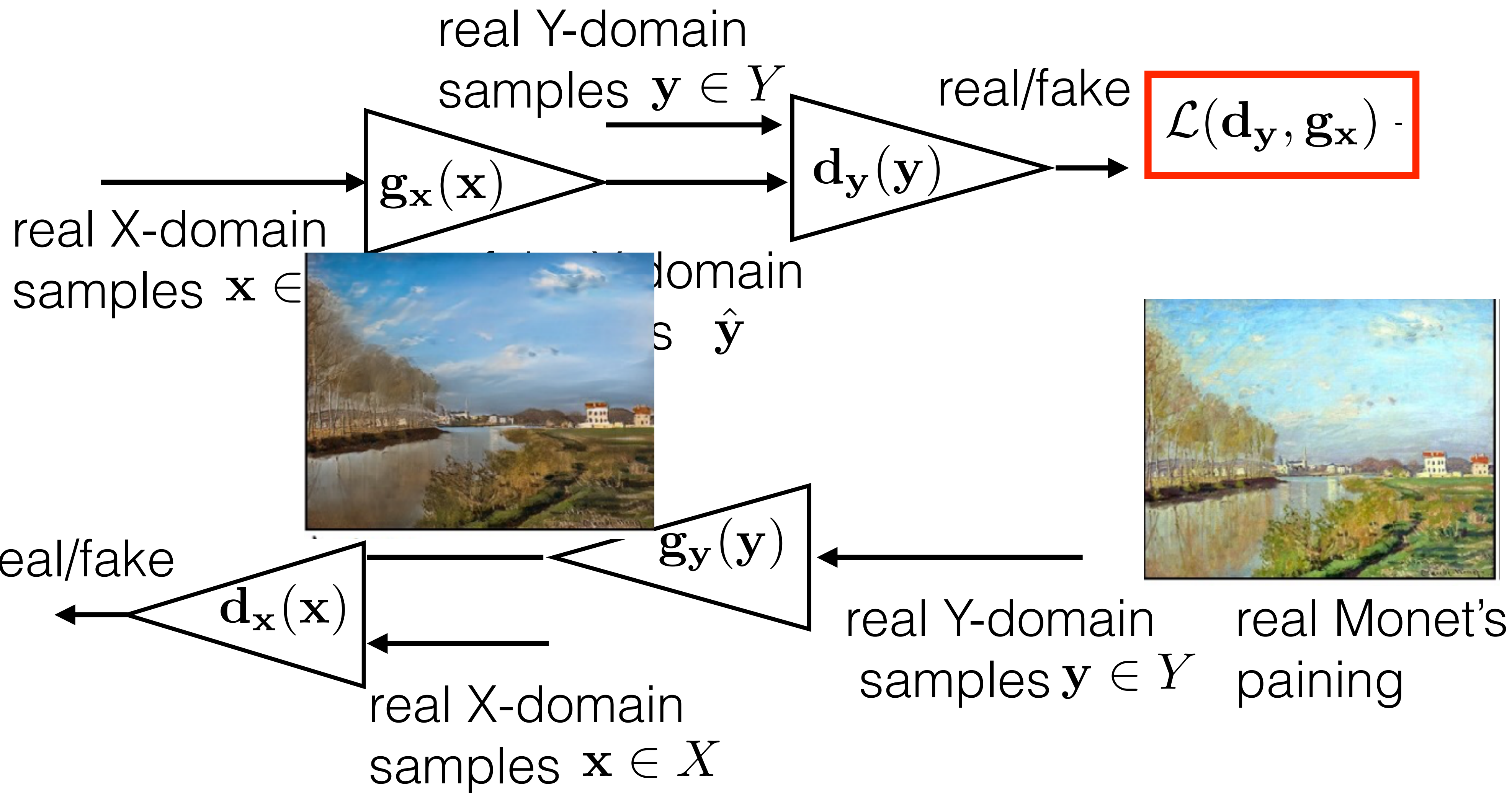
fake Monet's painting

Cycle-GAN [Zhu ICCV 2017]

<https://arxiv.org/abs/1703.10593>



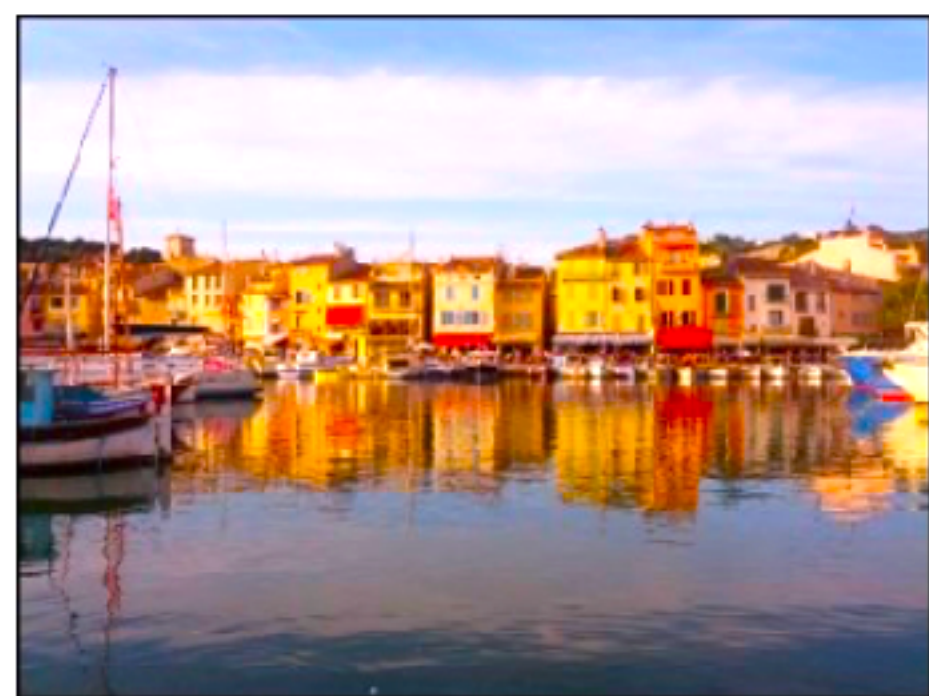
real Monet's painting



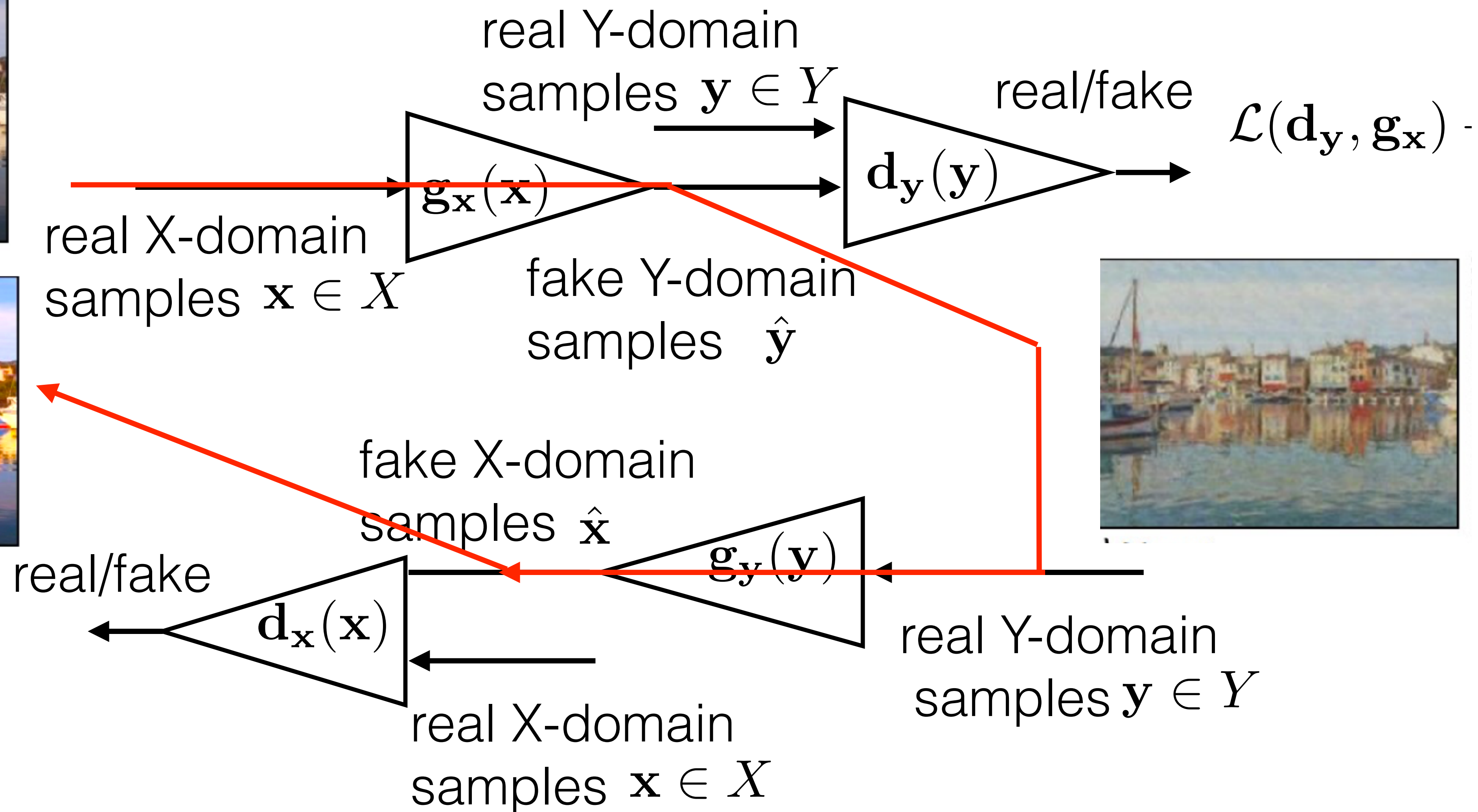
Cycle-GAN [Zhu ICCV 2017]

<https://arxiv.org/abs/1703.10593>

$$|\mathbf{g}_y(\mathbf{g}_x(\mathbf{x}) - \hat{\mathbf{x}}|$$



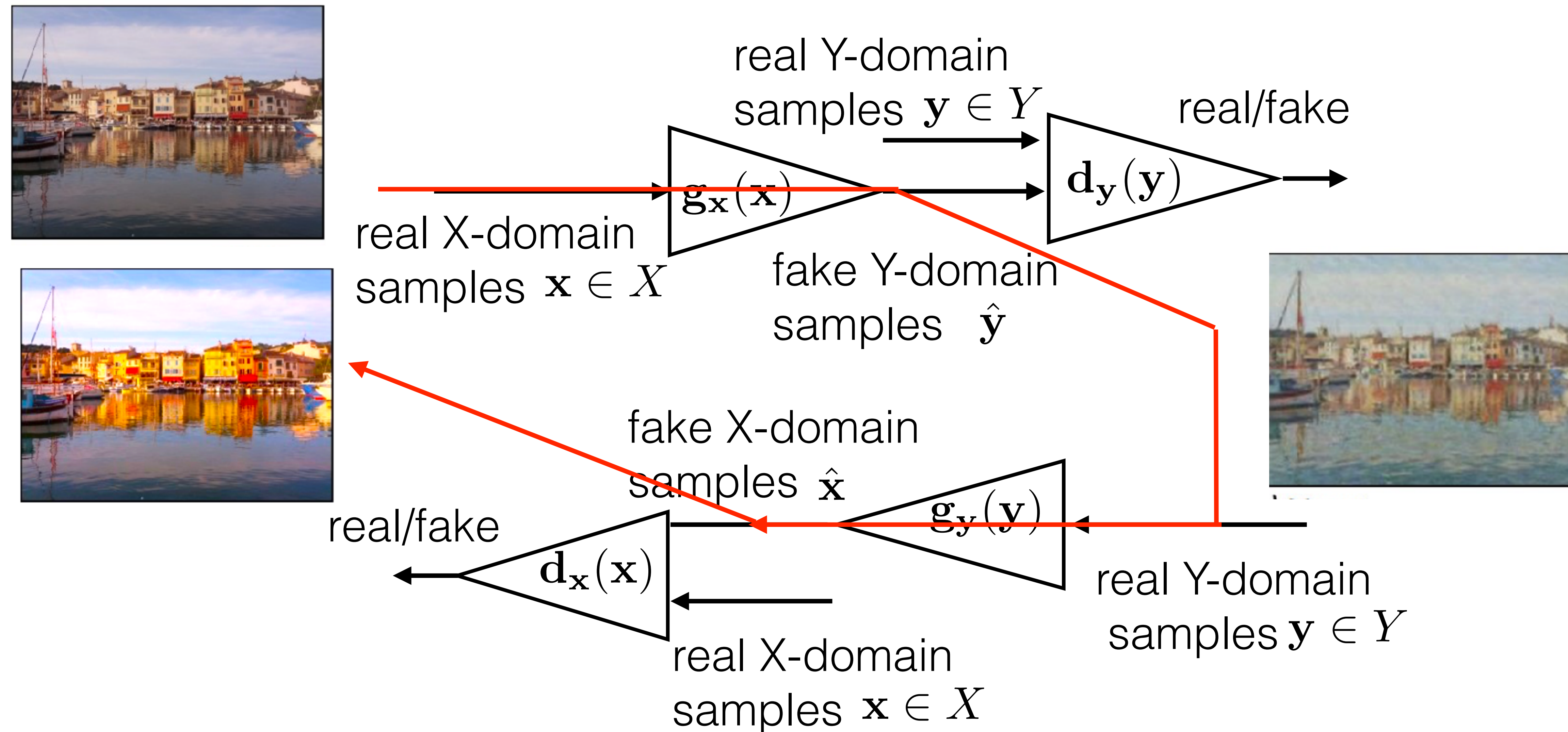
$$\mathcal{L}(\mathbf{d}_x, \mathbf{g}_y)$$



Cycle-GAN [Zhu ICCV 2017]

<https://arxiv.org/abs/1703.10593>

$$\mathcal{L}_{GAN}(\mathbf{d}_x, \mathbf{d}_y, \mathbf{g}_x, \mathbf{g}_y) = \mathcal{L}(\mathbf{d}_x, \mathbf{g}_y) + \mathcal{L}(\mathbf{d}_y, \mathbf{g}_x) + |\mathbf{g}_y(\mathbf{g}_x(\mathbf{x})) - \hat{\mathbf{x}}|$$



Cycle-GAN [Zhu ICCV 2017]
<https://arxiv.org/abs/1703.10593>

Monet $\leftrightarrow$ Photos



Monet $\rightarrow$ photo



photo $\rightarrow$ Monet

Cycle-GAN [Zhu ICCV 2017]
<https://arxiv.org/abs/1703.10593>

Zebras  **Horses**



zebra → horse



horse → zebra

Cycle-GAN [Zhu ICCV 2017]
<https://arxiv.org/abs/1703.10593>

Summer $\leftrightarrow$ Winter



summer $\rightarrow$ winter



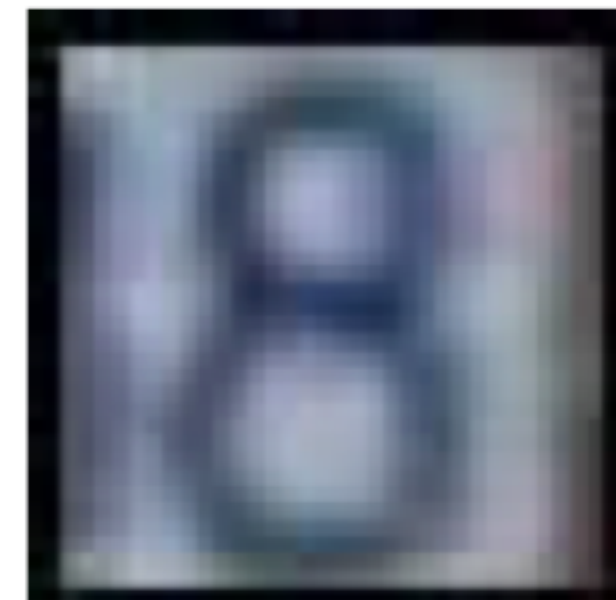
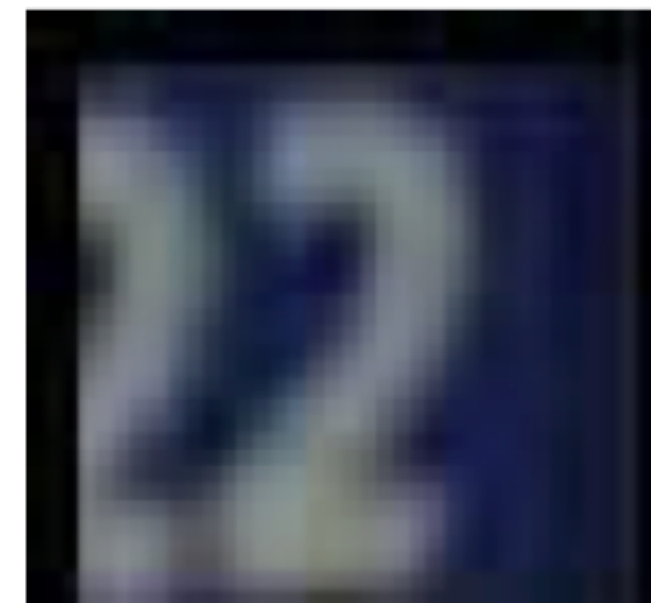
winter $\rightarrow$ summer

CyCaDa-GAN [Hoffman CVPR 2018]

<https://arxiv.org/pdf/1711.03213.pdf>

House numbers to MNIST transfer

y



House numbers

$$\mathcal{L}_{GAN}(\mathbf{d}_x, \mathbf{d}_y, \mathbf{g}_x, \mathbf{g}_y) = \mathcal{L}(\mathbf{d}_y, \mathbf{g}_x) + \mathcal{L}(\mathbf{d}_x, \mathbf{g}_y) + |\mathbf{g}_x(\mathbf{g}_y(y)) - \hat{y}|$$

- Cycle consistency helps, but it still allows to learn totally semantically inconsistent transfer

CyCaDa-GAN [Hoffman CVPR 2018]

<https://arxiv.org/pdf/1711.03213.pdf>

House numbers to MNIST transfer

y $g_y(y)$



House numbers MNIST

$$\mathcal{L}_{GAN}(\mathbf{d}_x, \mathbf{d}_y, \mathbf{g}_x, \mathbf{g}_y) = \mathcal{L}(\mathbf{d}_y, \mathbf{g}_x) + \mathcal{L}(\mathbf{d}_x, \mathbf{g}_y) + |\mathbf{g}_x(\mathbf{g}_y(y)) - \hat{y}|$$

- Cycle consistency helps, but it still allows to learn totally semantically inconsistent transfer

CyCaDa-GAN [Hoffman CVPR 2018]

<https://arxiv.org/pdf/1711.03213.pdf>

House numbers to MNIST transfer

y $g_y(y)$ $g_x(g_y(y))$



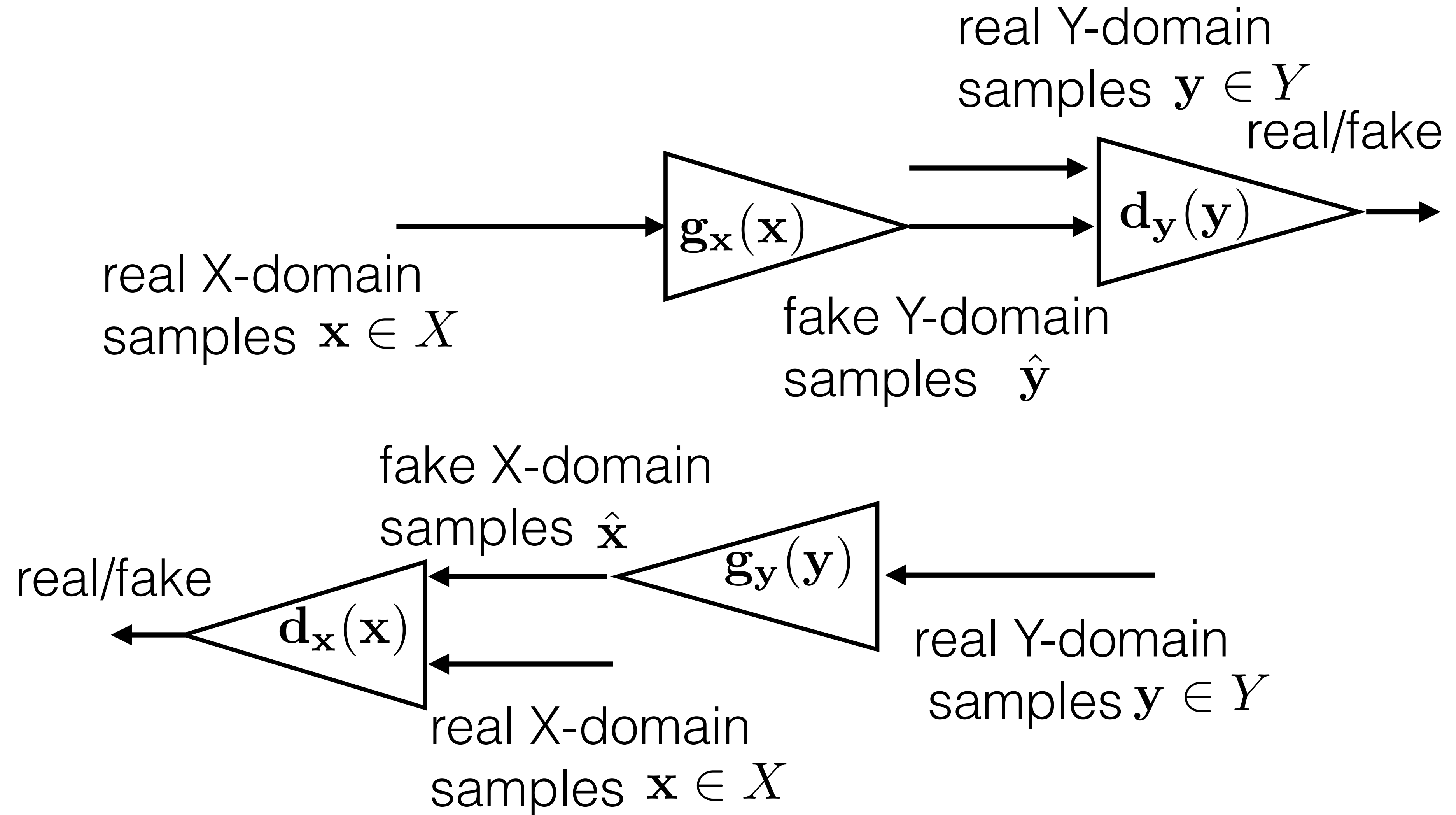
House numbers MNIST House numbers

$$\mathcal{L}_{GAN}(\mathbf{d}_x, \mathbf{d}_y, \mathbf{g}_x, \mathbf{g}_y) = \mathcal{L}(\mathbf{d}_y, \mathbf{g}_x) + \mathcal{L}(\mathbf{d}_x, \mathbf{g}_y) + |\mathbf{g}_x(\mathbf{g}_y(y)) - \hat{y}|$$

- Cycle consistency helps, but it still allows to learn totally semantically inconsistent transfer

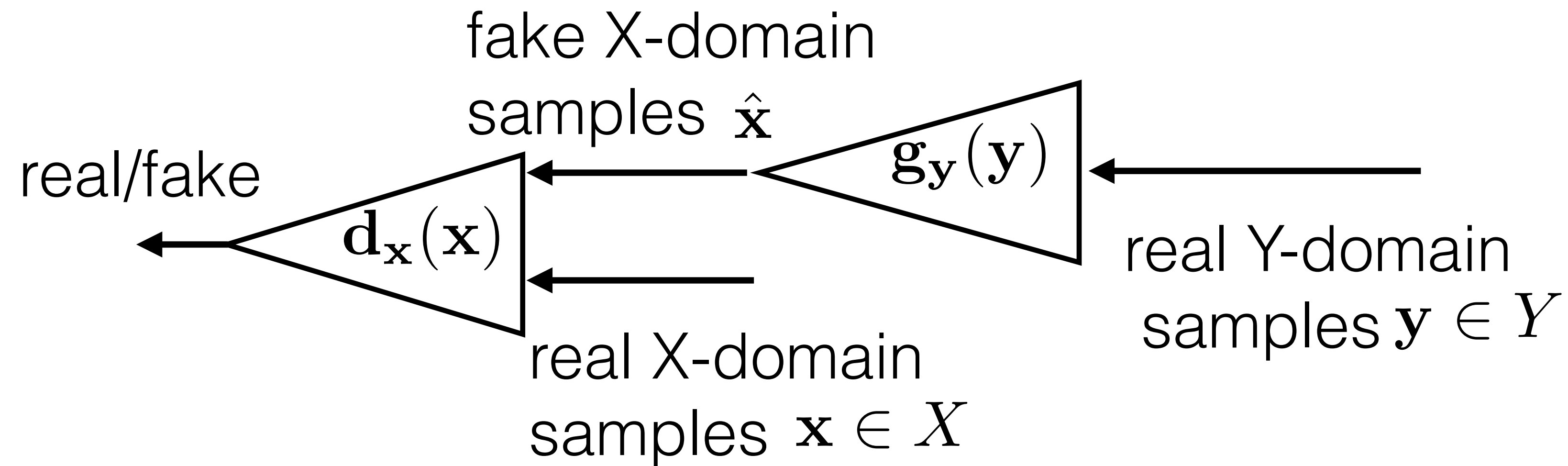
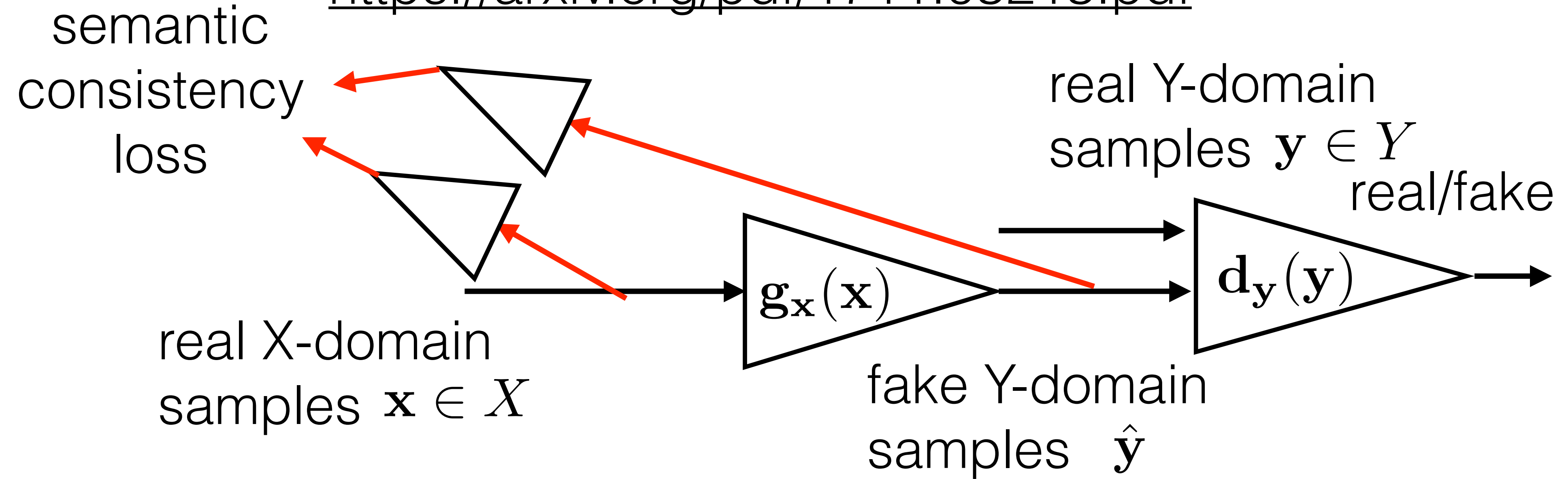
CyCaDa-GAN [Hoffman CVPR 2018]

<https://arxiv.org/pdf/1711.03213.pdf>



CyCaDa-GAN [Hoffman CVPR 2018]

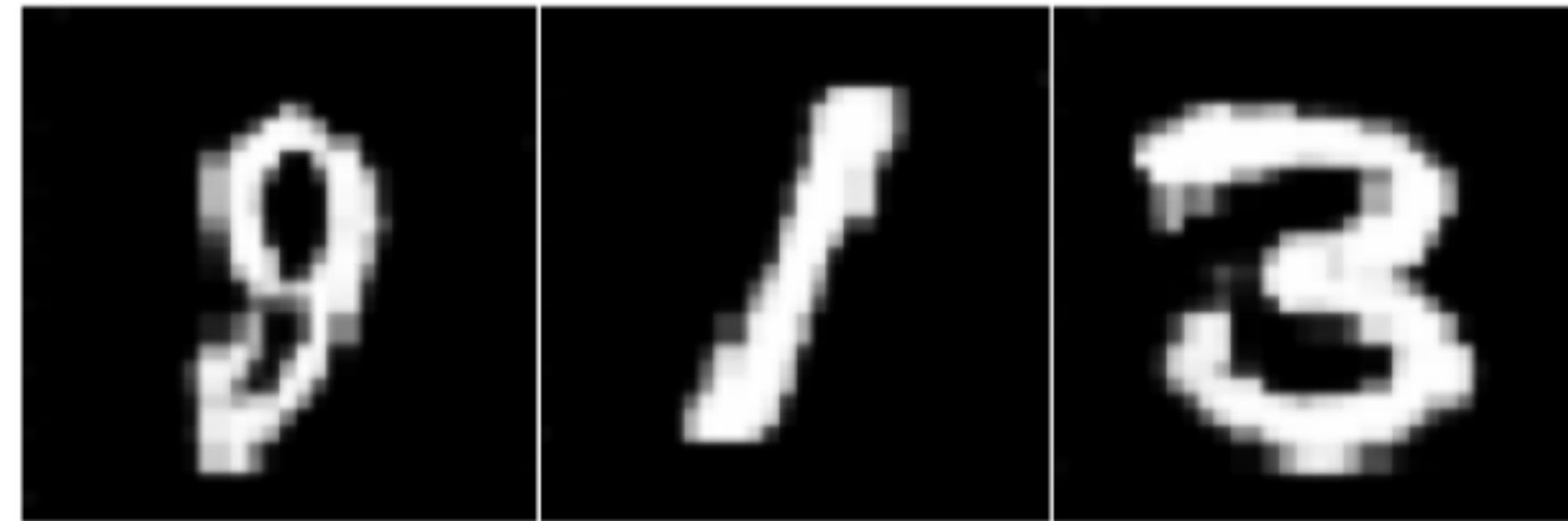
<https://arxiv.org/pdf/1711.03213.pdf>



- Semantic consistency enforce transformation to be semantically consistent



house numbers



MNIST

Diffusion models



Physics diffusion

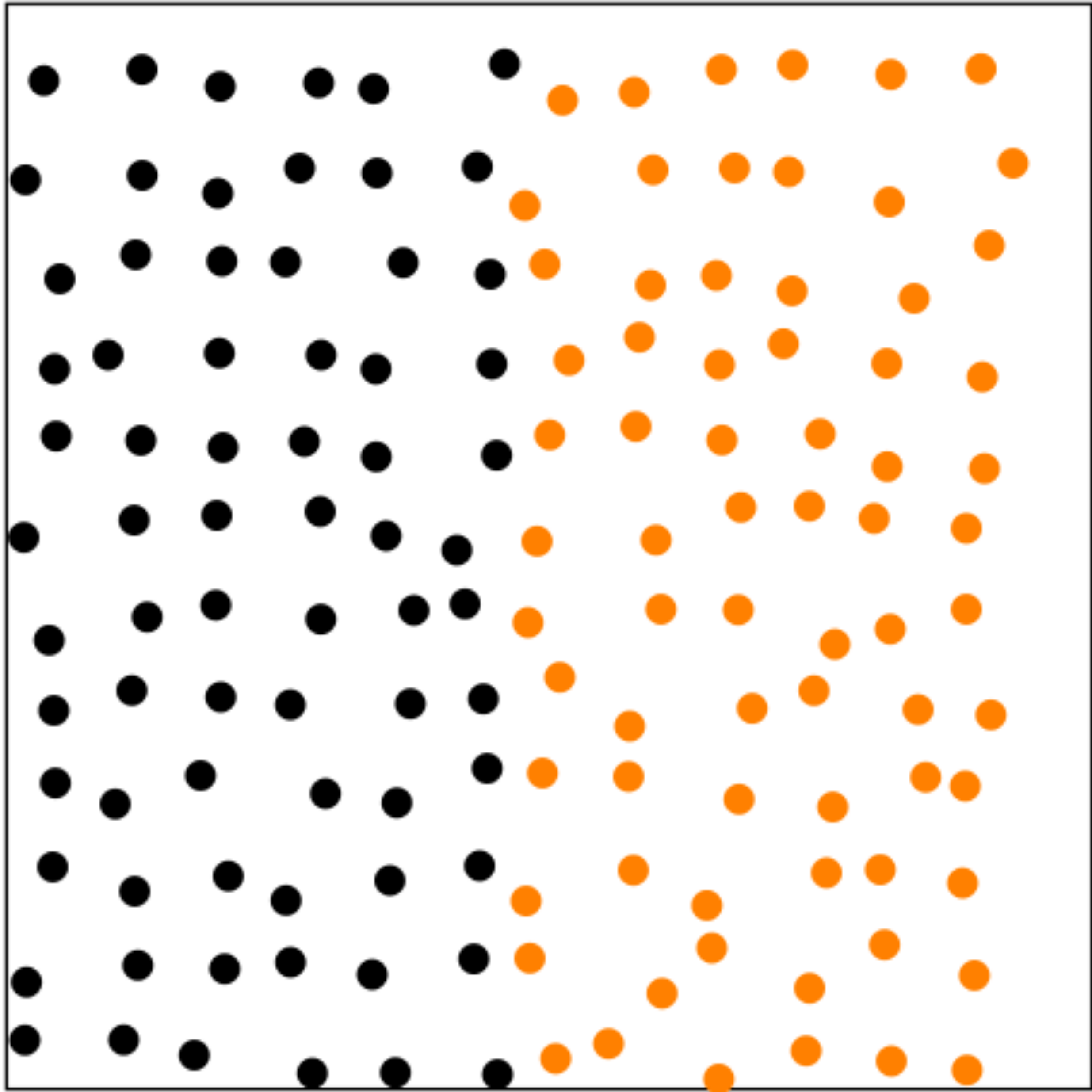
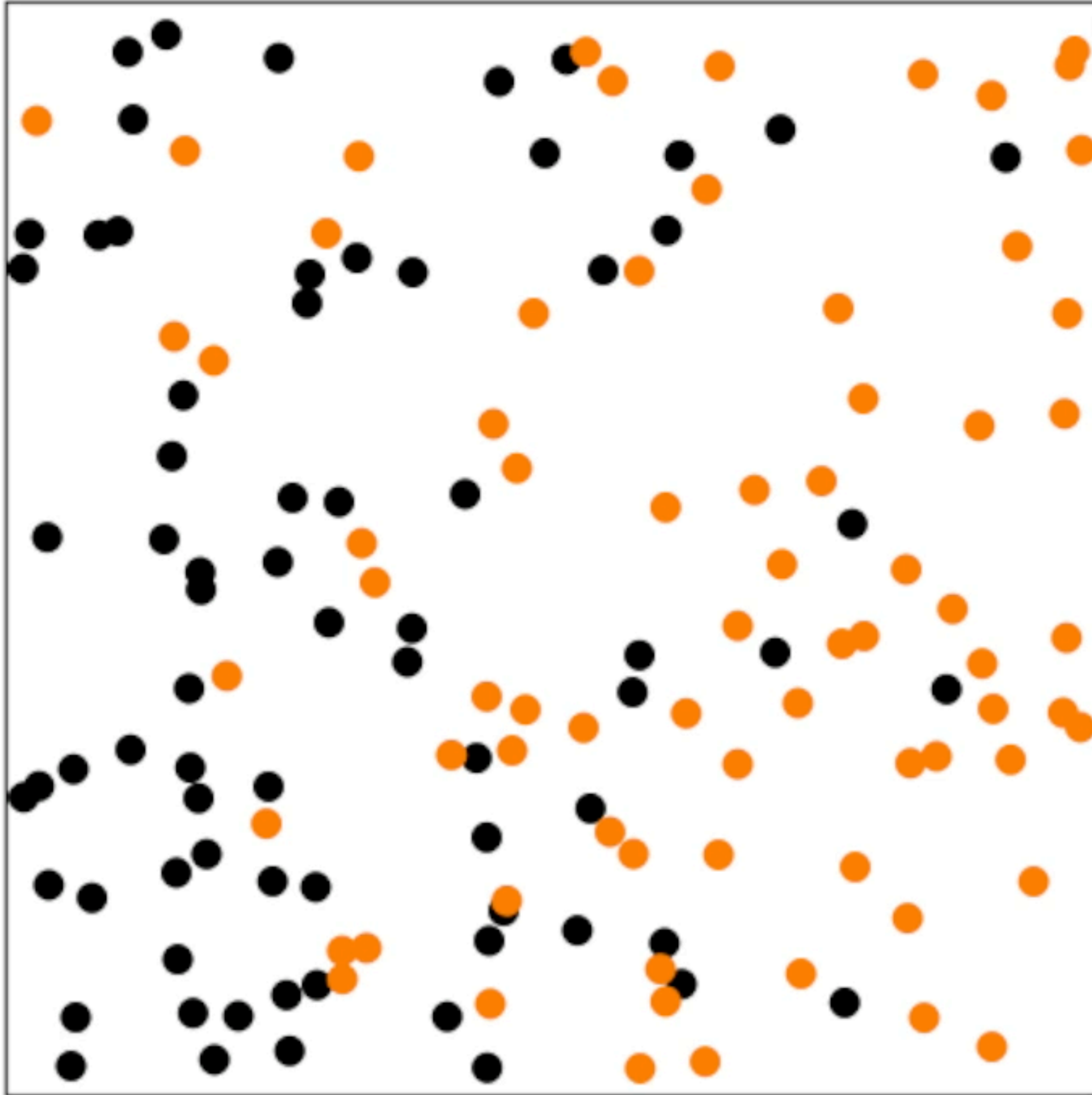


Image diffusion



Is the process reversible?

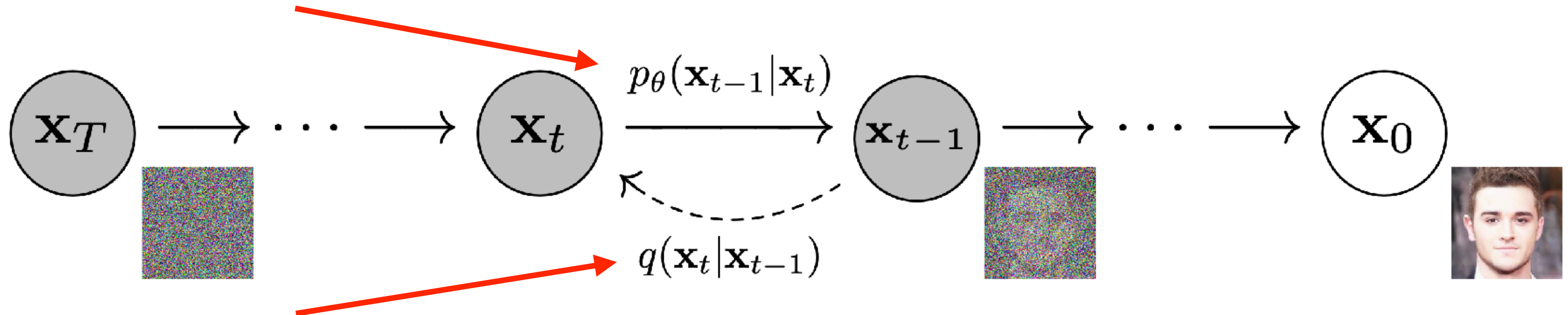
Diffusion models



Partially (if everything is known)

Diffusion models [Hu NeuriPS 2020]

Reverse of the diffusion process to generate original data from the noise



Markov chain of diffusion steps in which we slowly and randomly add noise to

$$\mathbf{x}_t = \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \epsilon$$

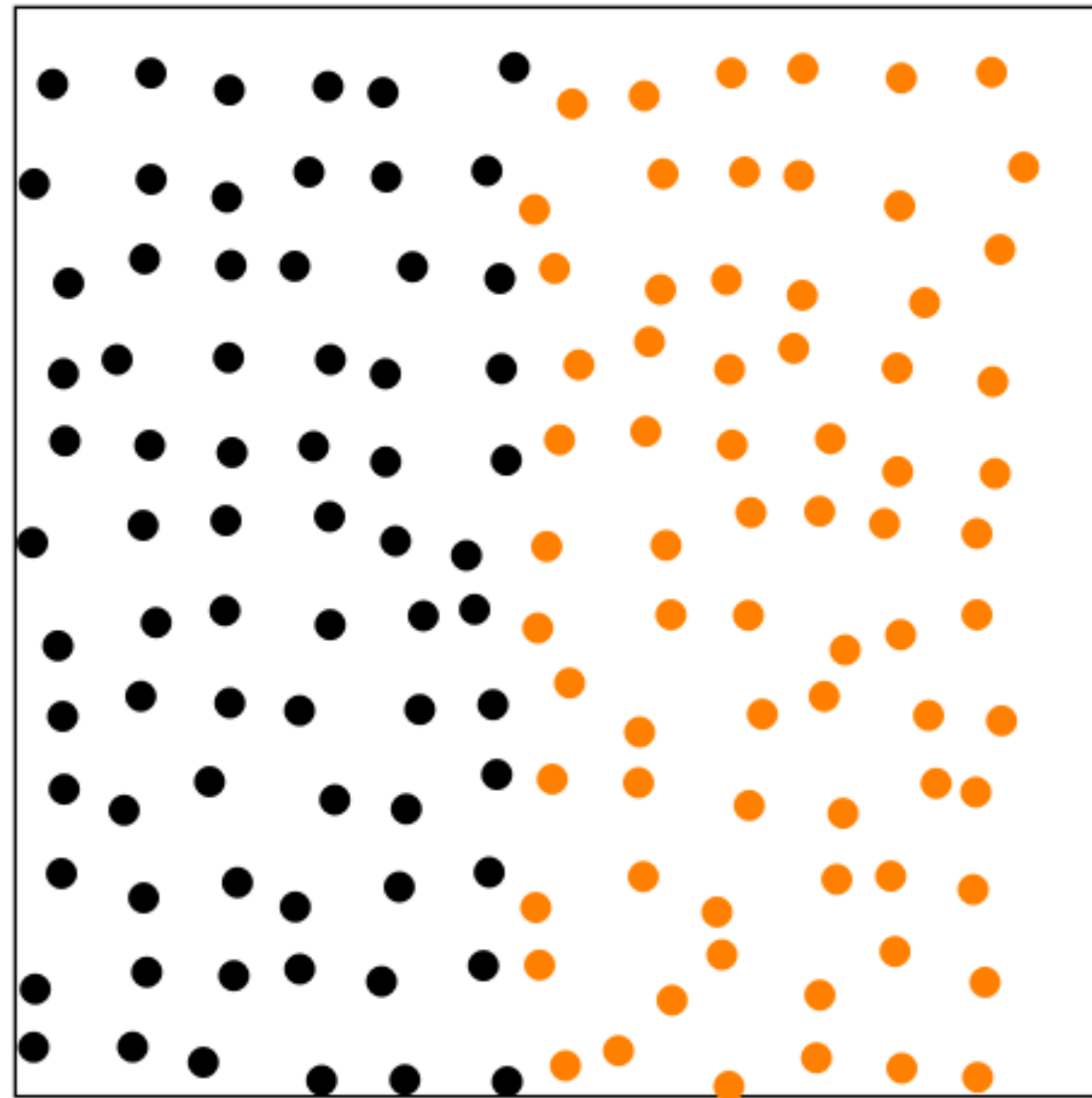
If noise is small the backward step has also “almost” gaussian distribution

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$$

=> learn de-noising networks through L2-norm

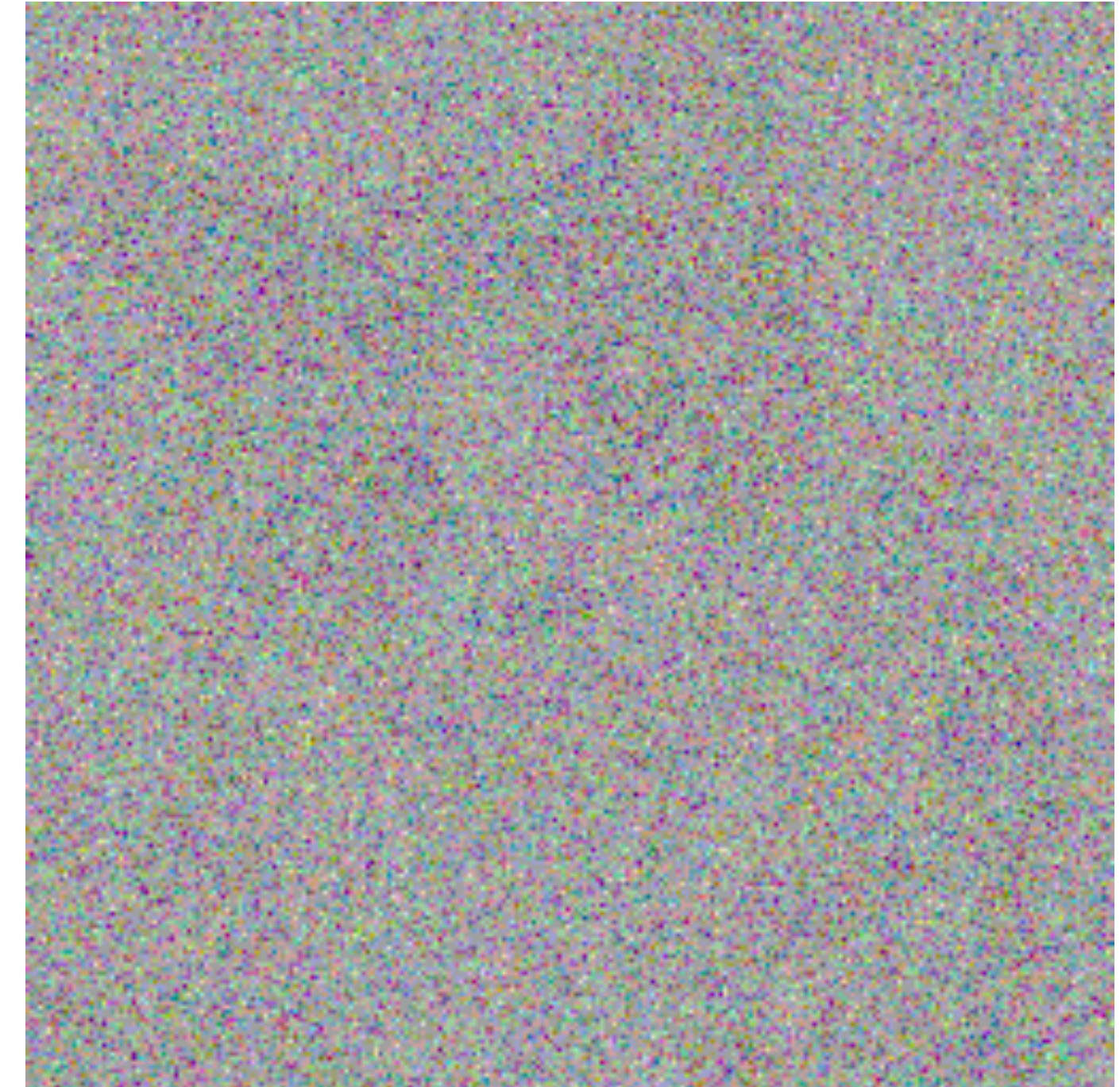
Diffusion models

Physics diffusion



$$\frac{\partial u}{\partial t} = D \nabla^2 u$$

Image diffusion



$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1 - \alpha_t} \epsilon$$

Stochastic

Entropy
increases

Partially
Reversible

Diffusion networks

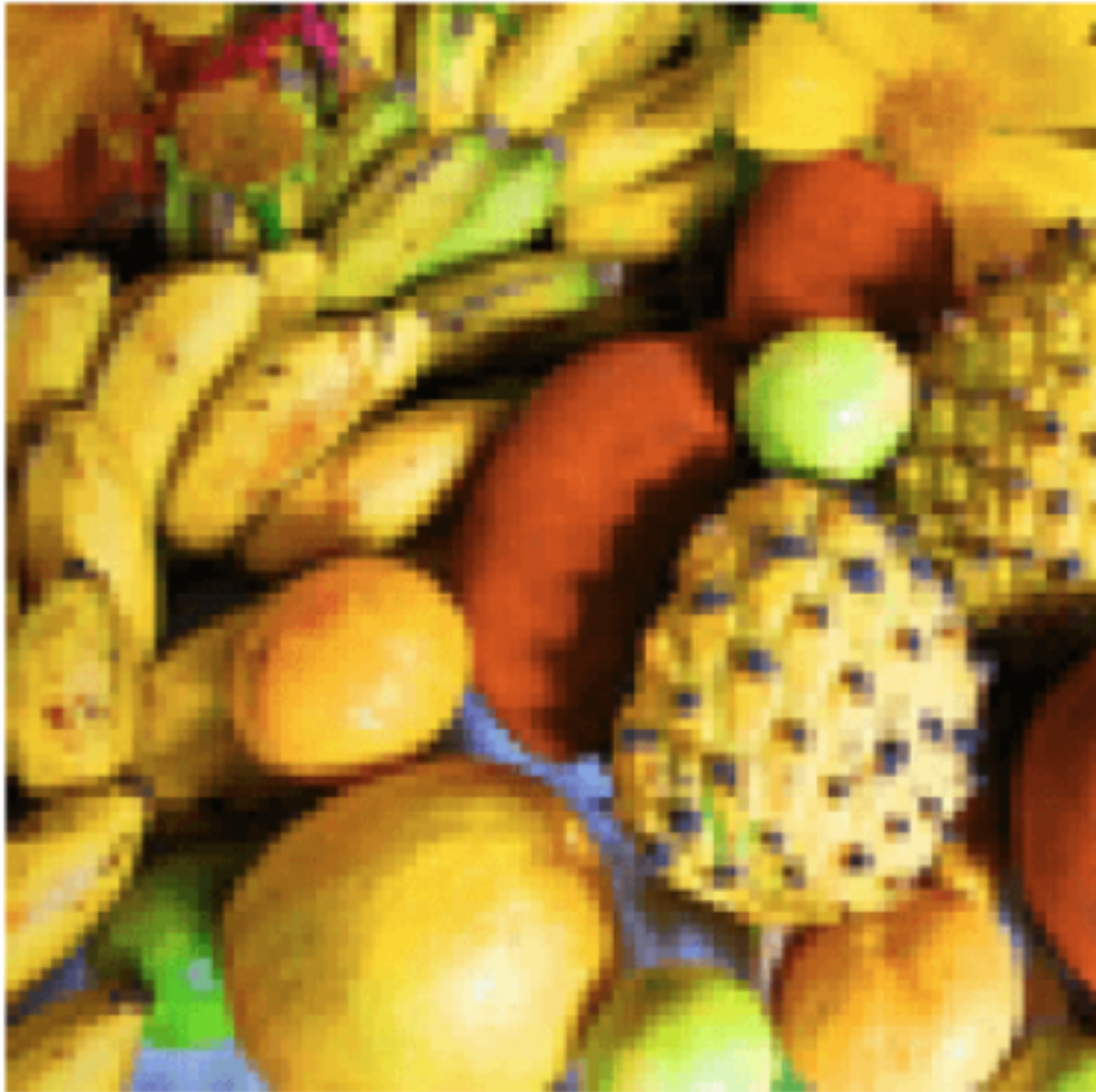


Jason Allen
Pueblo West
Théâtre D'opéra Spatial
\$750

Colorado State Fair

Super resolution

input



output



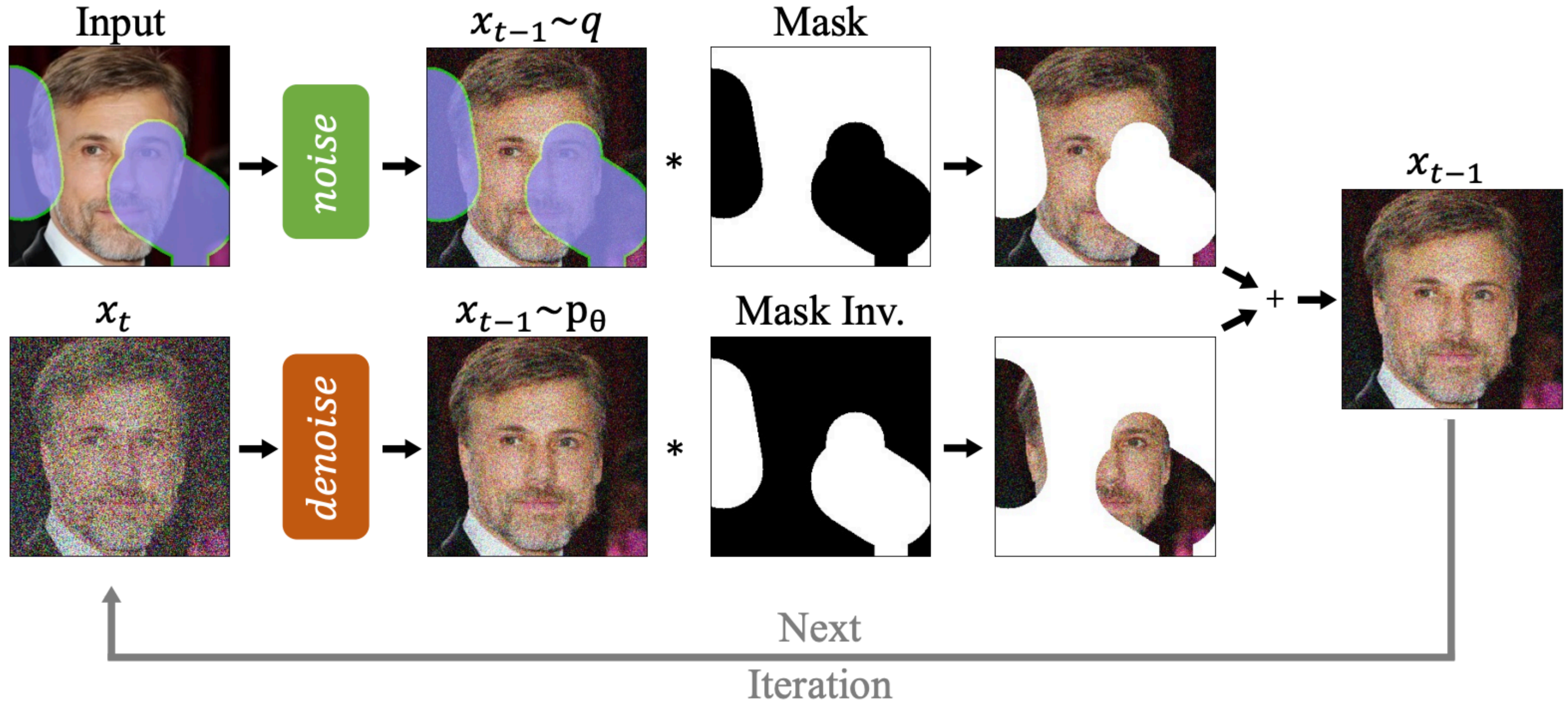
Inpainting



[Repaint, CVPR 2022]

https://openaccess.thecvf.com/content/CVPR2022/papers/Lugmayr_RePaint_Inpainting_Using_Denoising_Diffusion_Probabilistic_Models_CVPR_2022_paper.pdf

Inpainting



[Repaint, CVPR 2022]

Conditioning by style

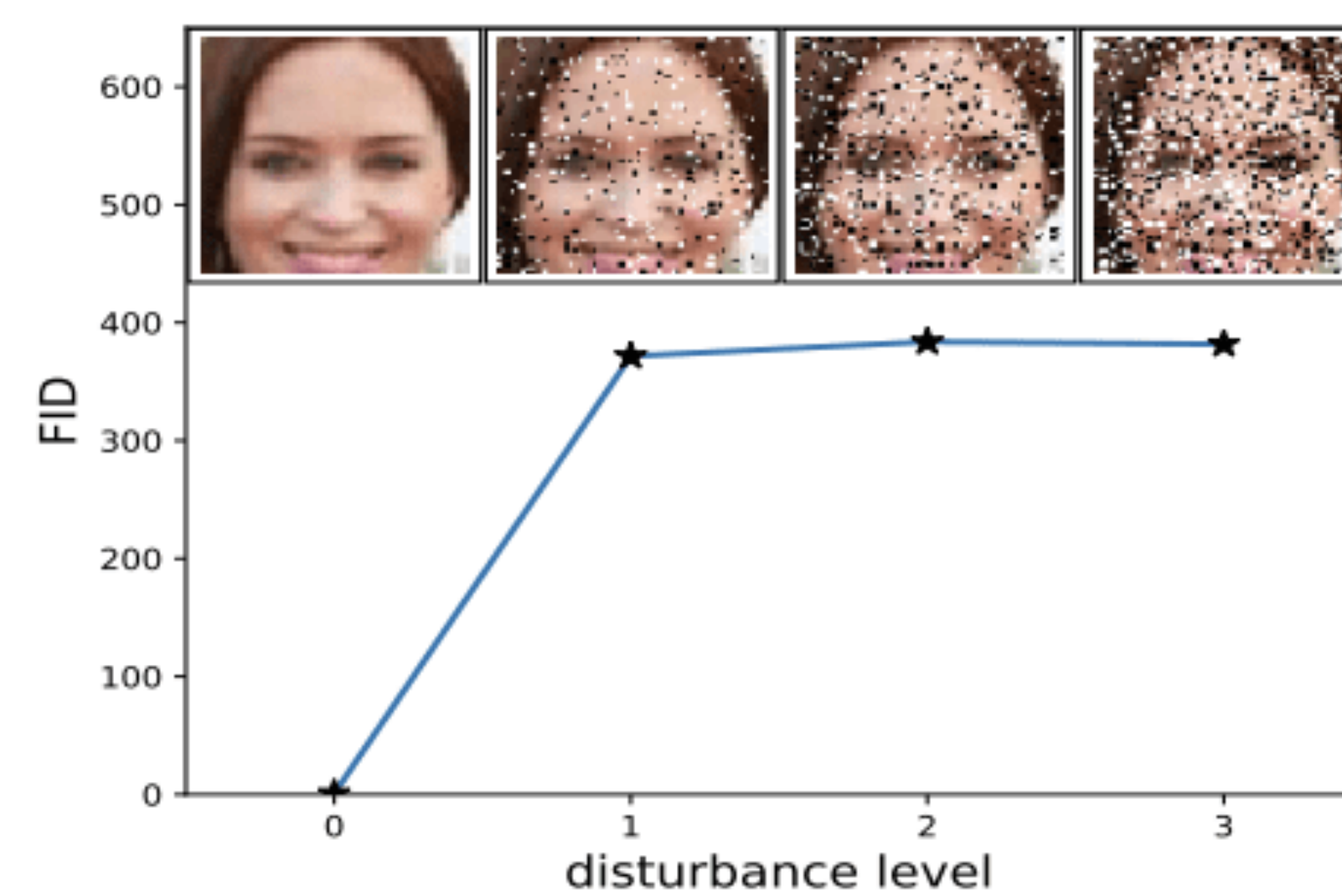
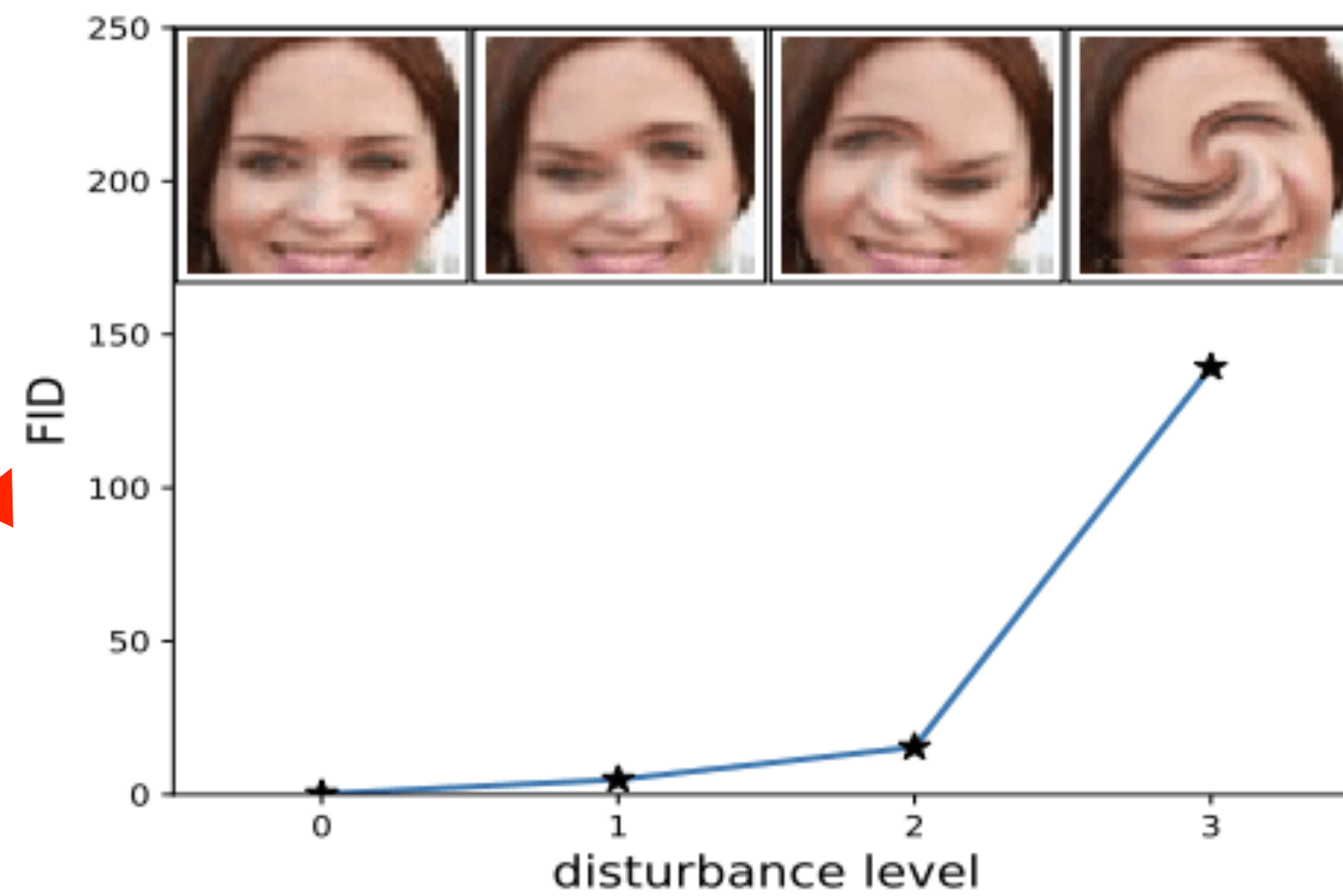
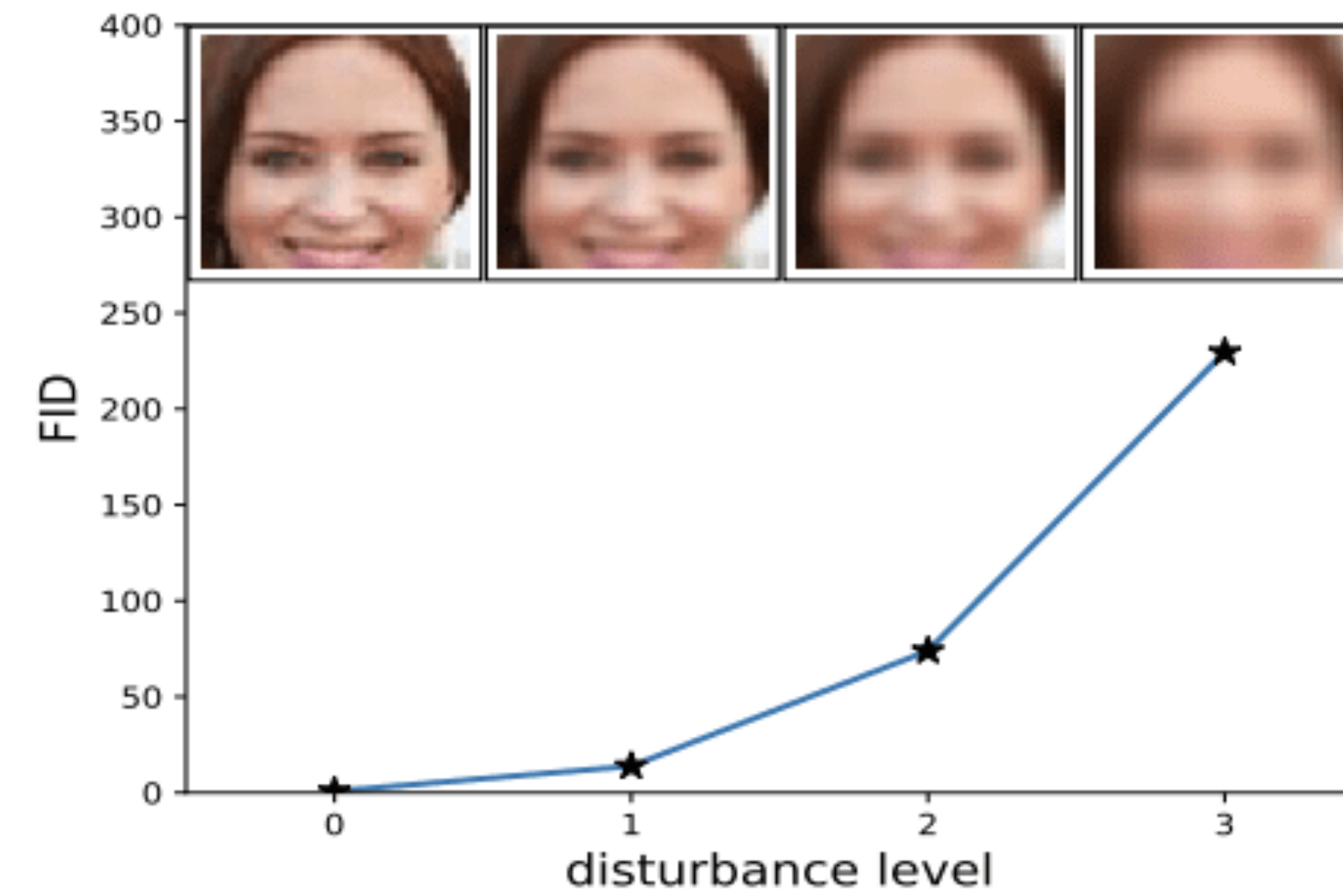
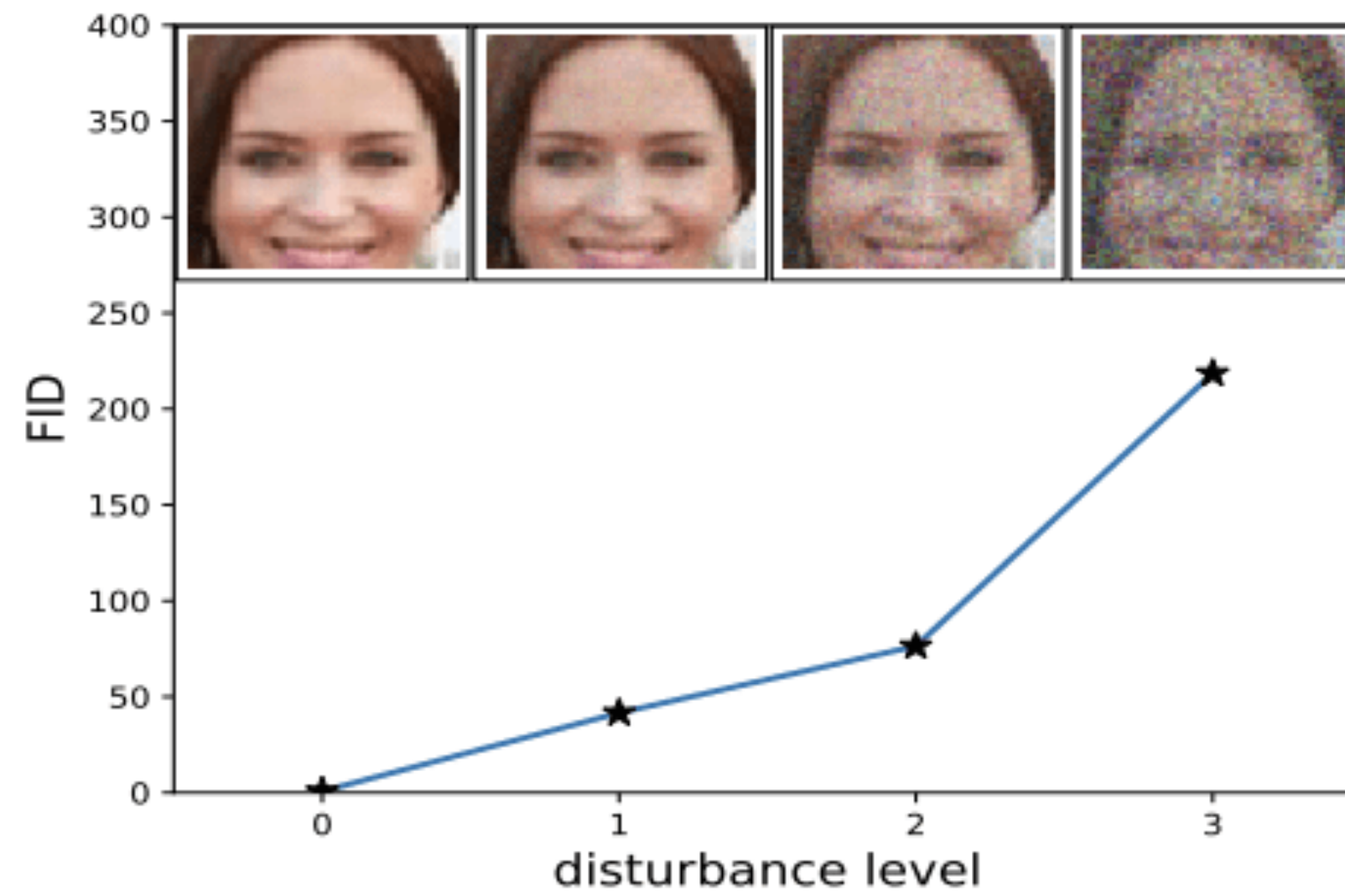


Conditioning by description



Brooks, Tim, Aleksander Holynski, and Alexei A. Efros. "Instructpix2pix: Learning to follow image editing instructions." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023.

Measuring quality of generated images???



Comparing mean + std on inception v3 in the final layer with real images

Data bias

input

output



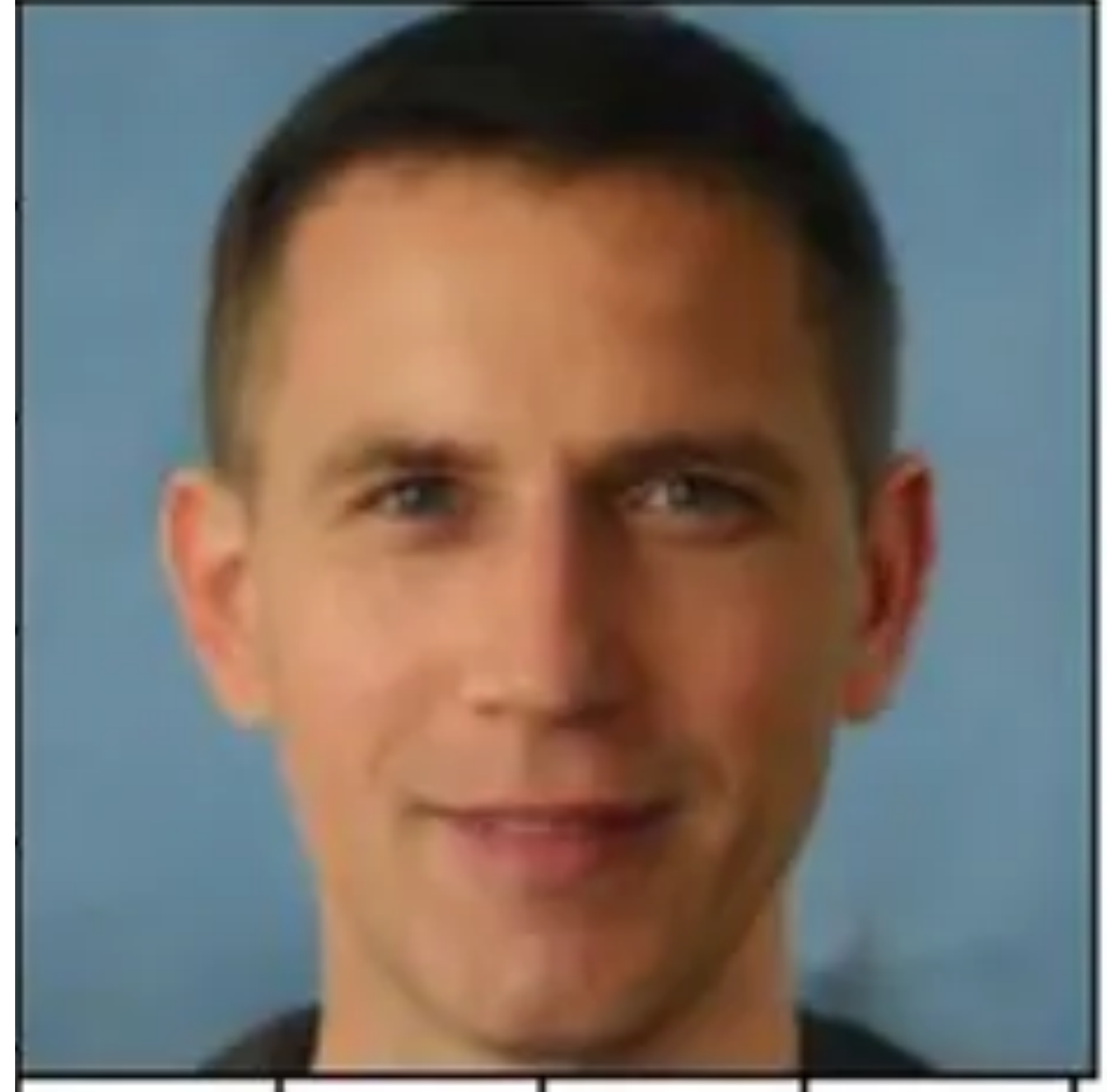
For the last in this course: How is it?

Data bias

input



output



For the last in this course: How is it?