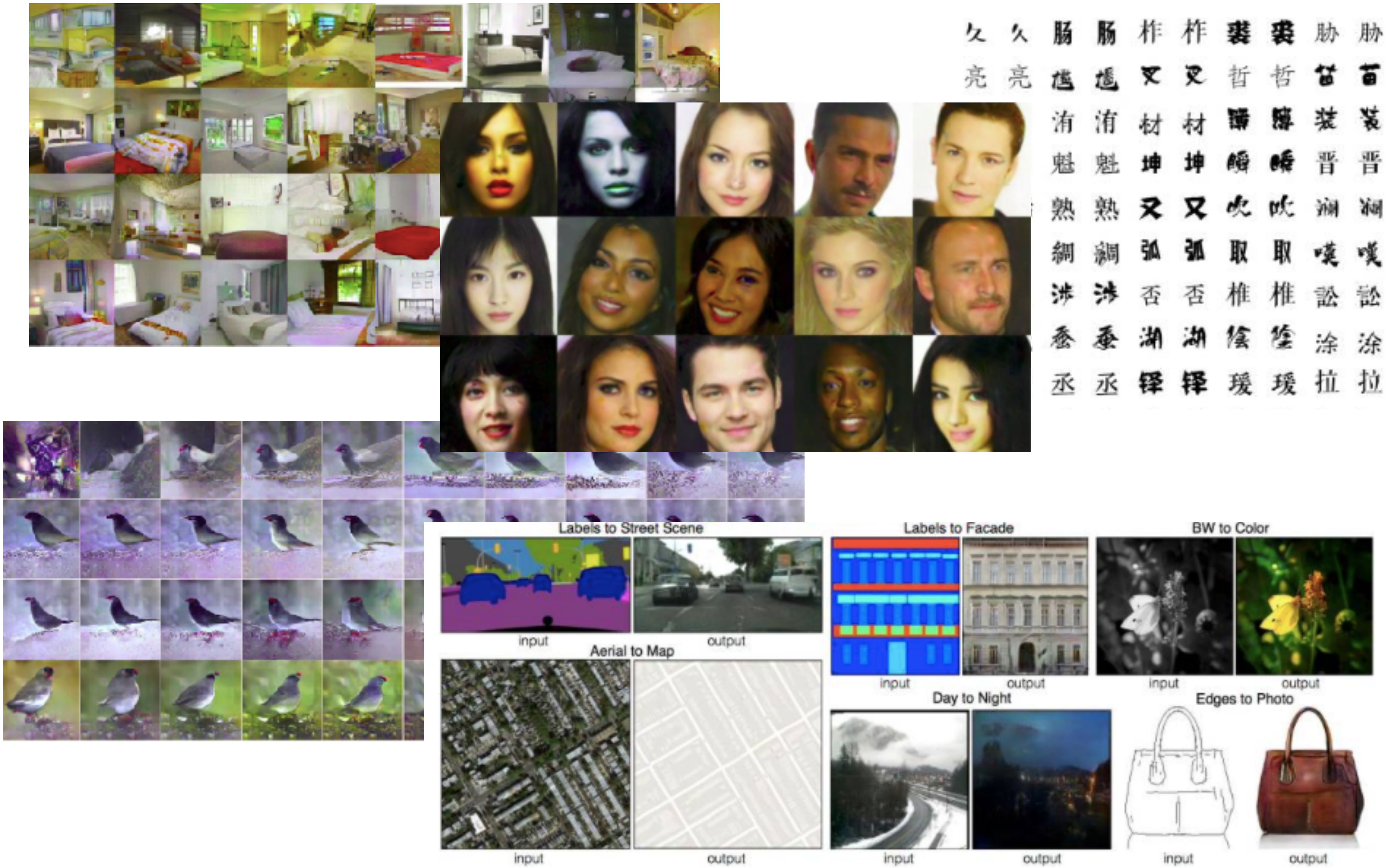
Wasserstein GAN

Martin Arjovsky[1], Soumith Chintala[2], and Léon Bottou[1,2]

[1]Courant Institute of Mathematical Sciences

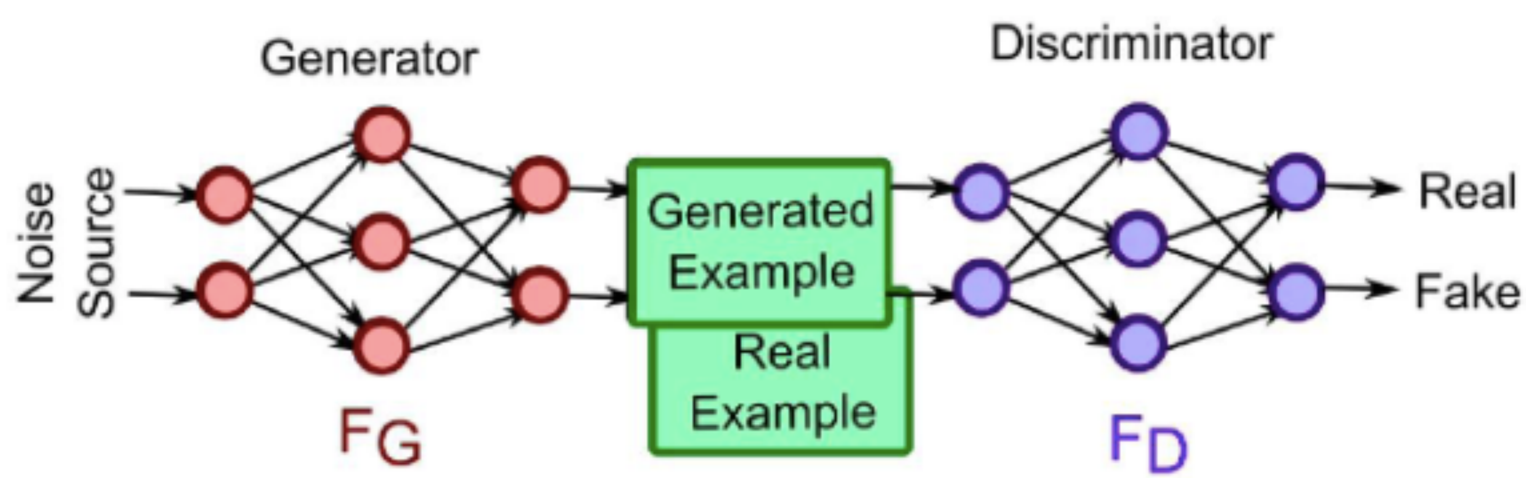
[2]Facebook AI Research

WHAT is GAN doing?



What is GAN ?

A min-max game between two components: generator G and discriminator D



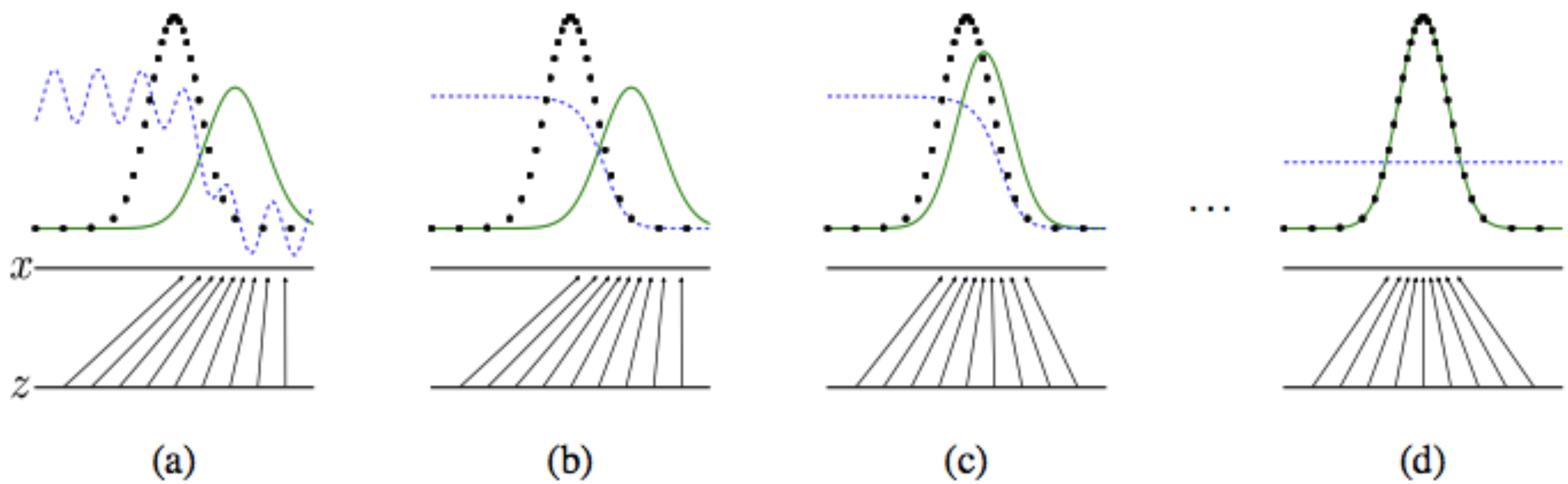
$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

There is two loss function for training generator:

$$\mathbb{E}_{x \sim P_g}[\log(1 - D(x))] \quad \textbf{(1)}$$

$$\mathbb{E}_{x \sim P_g}[-\log D(x)] \quad \textbf{(2)}$$

If everything goes well……



However……

What is the result of training？(For loss function 1)

$$\mathbb{E}_{x \sim P_g}[\log(1 - D(x))]$$

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

$$\begin{aligned}
C(G) =& \max_D V(G, D) \\
=& \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}[\log(1 - D_G^*(G(\boldsymbol{z})))] \\
=& \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))] \\
=& \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x} \sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] \\
=& -\log(4) + KL\left(p_{\text{data}} \middle\| \frac{p_{\text{data}} + p_g}{2}\right) + KL\left(p_g \middle\| \frac{p_{\text{data}} + p_g}{2}\right) \\
=& -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \| p_g\right)
\end{aligned}$$

What is the result of training ?

(For loss function 1)

$$\mathbb{E}_{x \sim P_g}[\log(1 - D(x))]$$

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

$$\mathbb{E}_{x \sim P_g}[\log(1 - D(x))]$$

$$= -\log(4) + KL\left(p_{\text{data}} \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right) + KL\left(p_g \left\| \frac{p_{\text{data}} + p_g}{2}\right.\right)$$

$$= -\log(4) + 2 \cdot JSD\left(p_{\text{data}} \| p_g\right)$$

Problem

—— Pr and Pg are usually low-dimension manifold in high-dimension space. ==>

—— The measure of the overlapping portion of support set of Pr and Pg is 0. ==>

—— JSD(Pr||Pg) = log2, which is a constant. ==>

—— So gradient would be 0.

Finally, the gradient will vanish if discriminator is well-trained and the gradient is unstable if discriminator is not well-trained.

What is the result of training ?

(For loss function 2)

$$\mathbb{E}_{x \sim P_g}[-\log D(x)]$$

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

$$\mathbb{E}_{x \sim P_r}[\log D^*(x)] + \mathbb{E}_{x \sim P_g}[\log(1 - D^*(x))] = 2JS(P_r||P_g) - 2\log 2$$

$$\begin{aligned}
KL(P_g||P_r) &= \mathbb{E}_{x \sim P_g}[\log \frac{P_g(x)}{P_r(x)}] \\
&= \mathbb{E}_{x \sim P_g}[\log \frac{P_g(x)/(P_r(x) + P_g(x))}{P_r(x)/(P_r(x) + P_g(x))}] \\
&= \mathbb{E}_{x \sim P_g}[\log \frac{1 - D^*(x)}{D^*(x)}] \\
&= \mathbb{E}_{x \sim P_g} \log[1 - D^*(x)] - \mathbb{E}_{x \sim P_g} \log D^*(x)
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}_{x \sim P_g}[-\log D^*(x)] &= KL(P_g||P_r) - \mathbb{E}_{x \sim P_g} \log[1 - D^*(x)] \\
&= KL(P_g||P_r) - 2JS(P_r||P_g) + 2\log 2 + \mathbb{E}_{x \sim P_r}[\log D^*(x)]
\end{aligned}$$

What is the result of training ?
(For loss function 2)

$$\mathbb{E}_{x \sim P_g}[-\log D(x)]$$

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

$$\begin{aligned} \mathbb{E}_{x \sim P_g}[-\log D^*(x)] &= KL(P_g || P_r) - \mathbb{E}_{x \sim P_g} \log[1 - D^*(x)] \\ &= KL(P_g || P_r) - 2JS(P_r || P_g) + 2\log 2 + \mathbb{E}_{x \sim P_r}[\log D^*(x)] \end{aligned}$$

Problem

—— We are going to minimize KL divergence and maximize JS divergence at the same time

==> Gradient is unstable.

—— KL divergence is not symmetric.

==> Mode collapse.

Conclusion

- 1. Pr and Pg share negligibly same support set.

  ==>Add Noise.

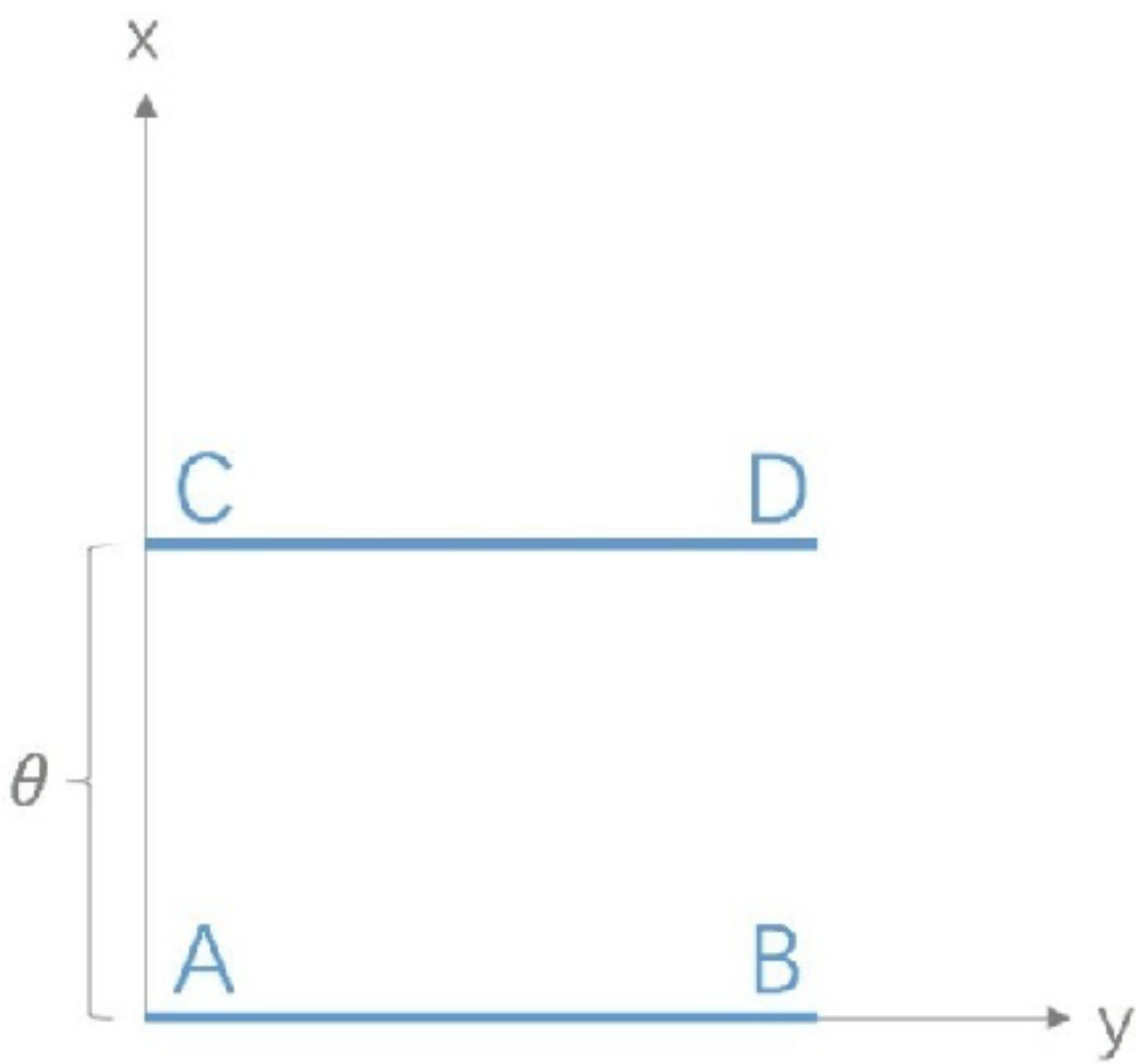
- 2.KL-divergence and JS-divergence are not suitable in this problem for training.

  ==>Wasserstein metric.

Wasserstein metric

Earth Mover Distance

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma}\big[\, \|x - y\| \,\big]$$

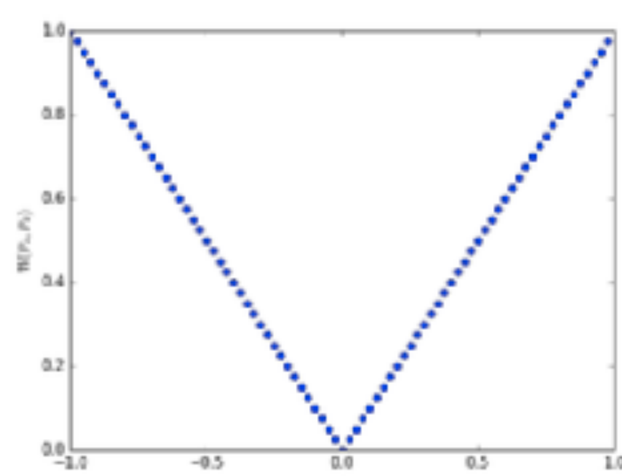


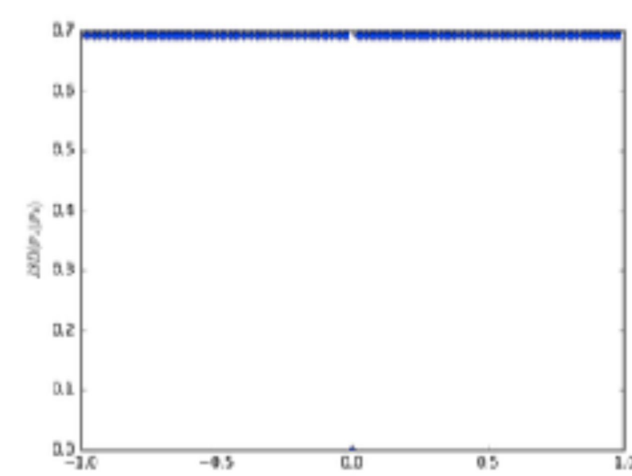
$$W(\mathbb{P}_0, \mathbb{P}_\theta) = |\theta|,$$

$$JS(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} \log 2 & \text{if } \theta \neq 0 \,, \\ 0 & \text{if } \theta = 0 \,, \end{cases}$$

$$KL(\mathbb{P}_\theta \| \mathbb{P}_0) = KL(\mathbb{P}_0 \| \mathbb{P}_\theta) = \begin{cases} +\infty & \text{if } \theta \neq 0 \,, \\ 0 & \text{if } \theta = 0 \,, \end{cases}$$

$$\text{and } \delta(\mathbb{P}_0, \mathbb{P}_\theta) = \begin{cases} 1 & \text{if } \theta \neq 0 \,, \\ 0 & \text{if } \theta = 0 \,. \end{cases}$$





Wasserstein metric

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \big[\|x - y\| \big]$$

By Kantorovich-Rubinstein duality

$$W(P_r, P_g) = \frac{1}{K} \sup_{||f||_L \leq K} \mathbb{E}_{x\sim P_r}[f(x)] - \mathbb{E}_{x\sim P_g}[f(x)]$$

$$K \cdot W(P_r, P_g) \approx \max_{w:|f_w|_L \leq K} \mathbb{E}_{x\sim P_r}[f_w(x)] - \mathbb{E}_{x\sim P_g}[f_w(x)]$$

Discriminator Loss: $\mathbb{E}_{x\sim P_g}[f_w(x)] - \mathbb{E}_{x\sim P_r}[f_w(x)]$

Generator Loss: $-\mathbb{E}_{x\sim P_g}[f_w(x)]$

Discriminator Gradients: $\nabla_w \left[\frac{1}{m}\sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))\right]$

Generator Gradients: $\nabla_\theta W(\mathbb{P}_r, \mathbb{P}_\theta) = -\mathbb{E}_{z\sim p(z)}[\nabla_\theta f(g_\theta(z))]$

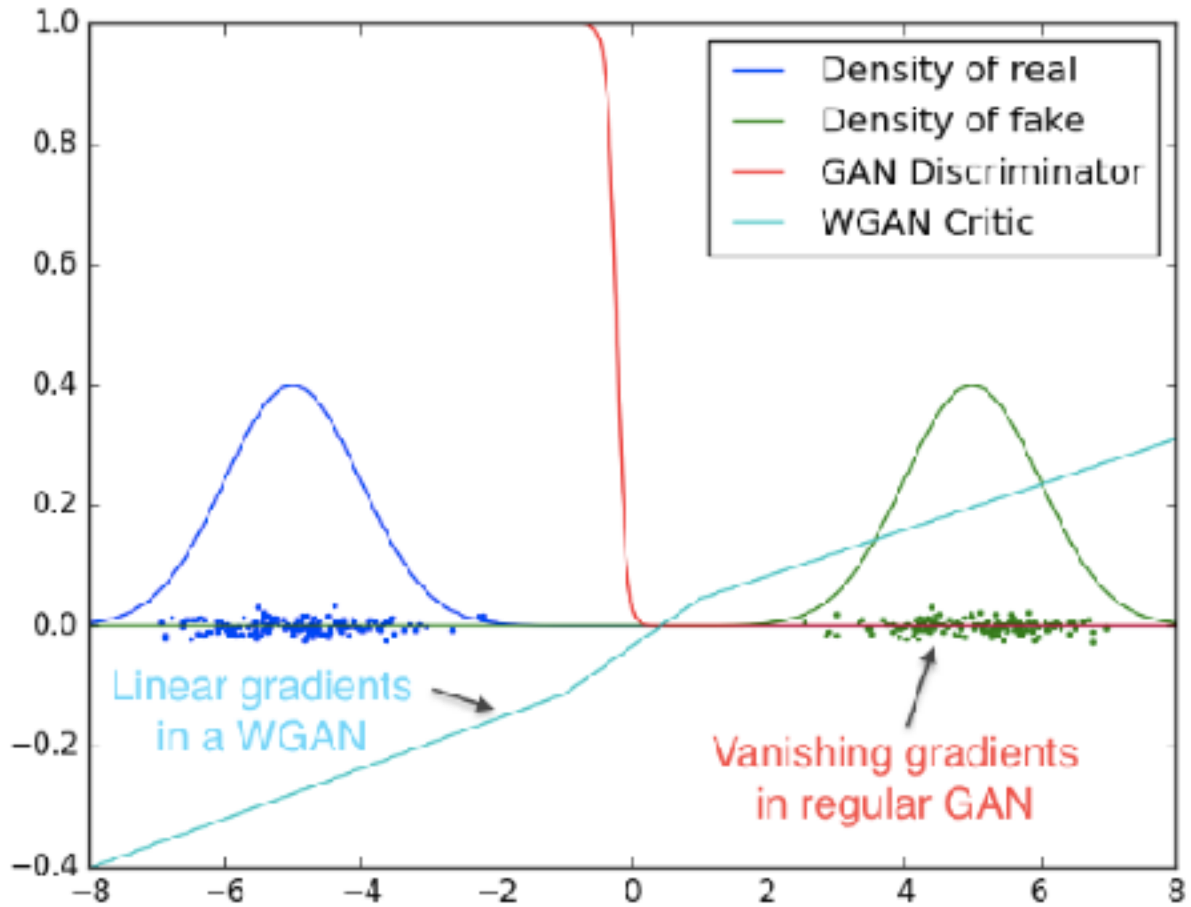
WGAN Training

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.

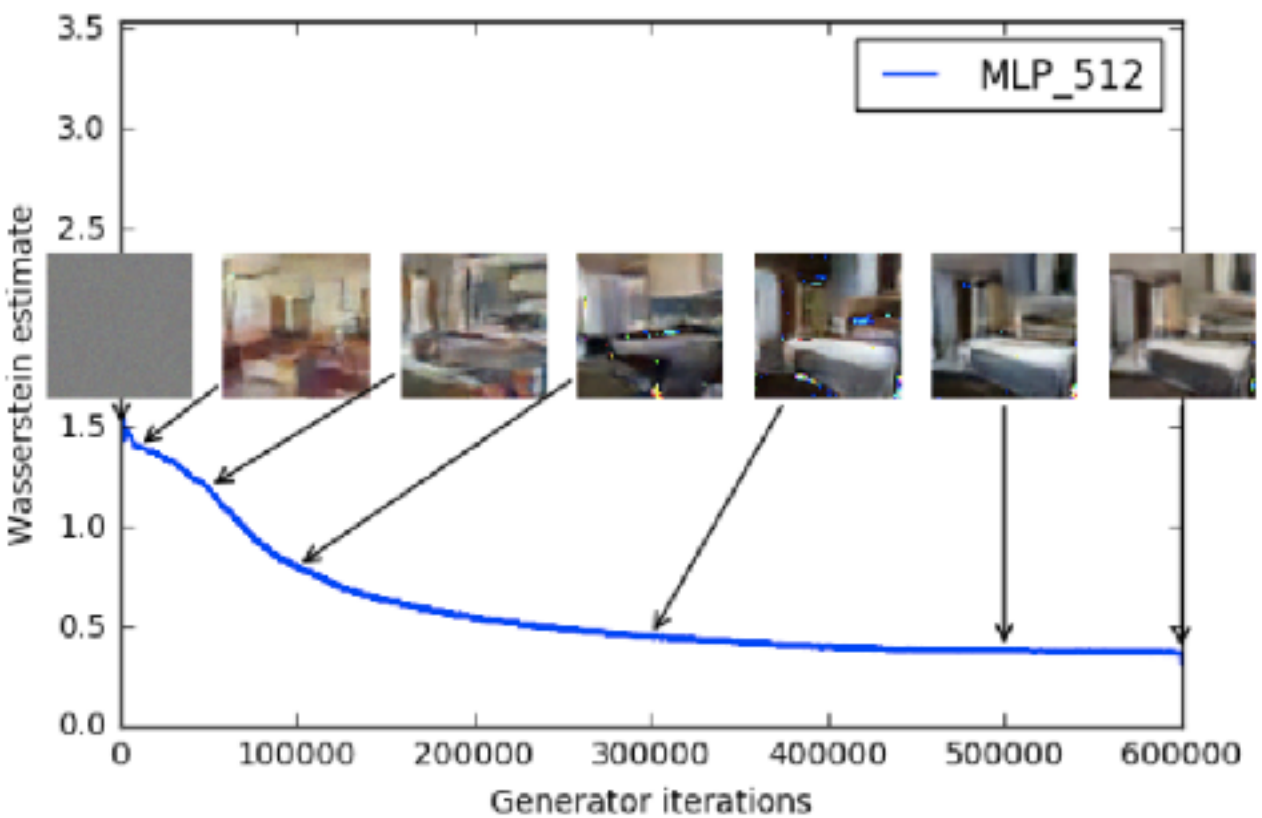
while θ has not converged **do**
    for $t = 0, ..., n_{\text{critic}}$ **do**
        Sample $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$ a batch from the real data.
        Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
        $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))\right]$
        $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
        $w \leftarrow \text{clip}(w, -c, c)$
    end for
    Sample $\{z^{(i)}\}_{i=1}^m \sim p(z)$ a batch of prior samples.
    $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^m f_w(g_\theta(z^{(i)}))$
    $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
end while

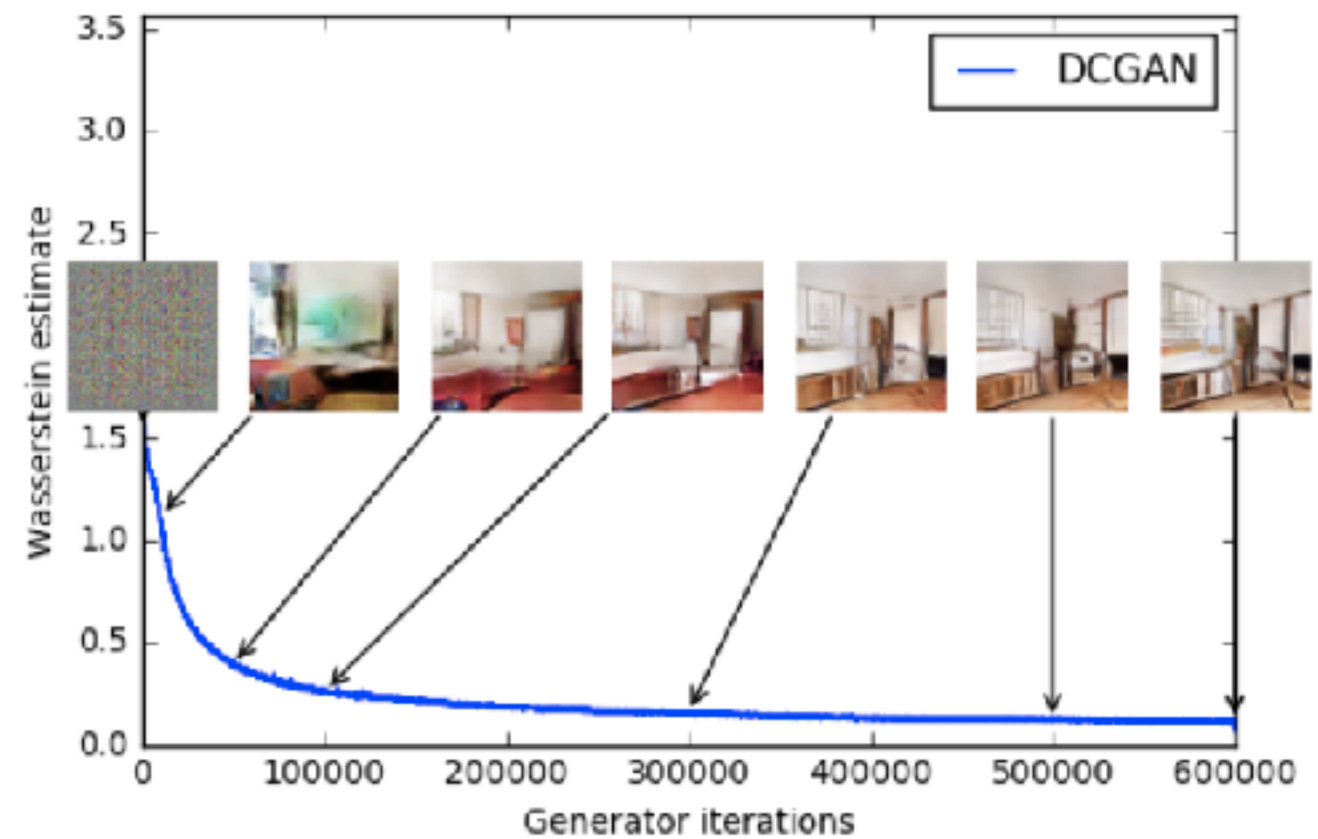
Result



WGAN Critic would keep linear gradients almost everywhere.
No Gradient vanishing problem.

Result





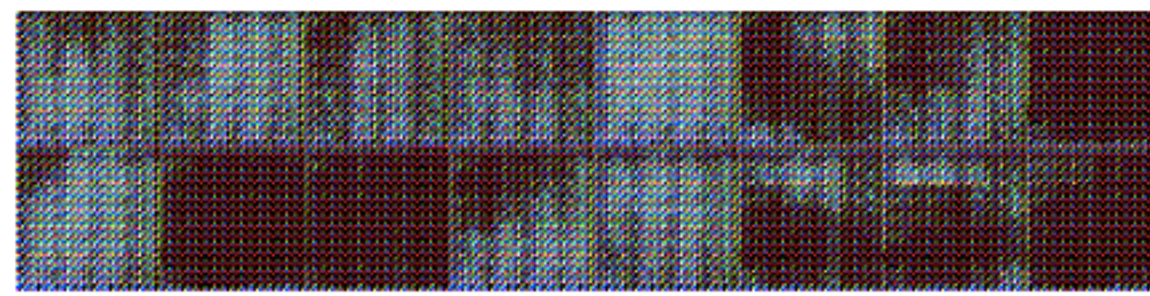
Wasserstein metrics is a good metric for this problem.
The less value, the better image.

Result

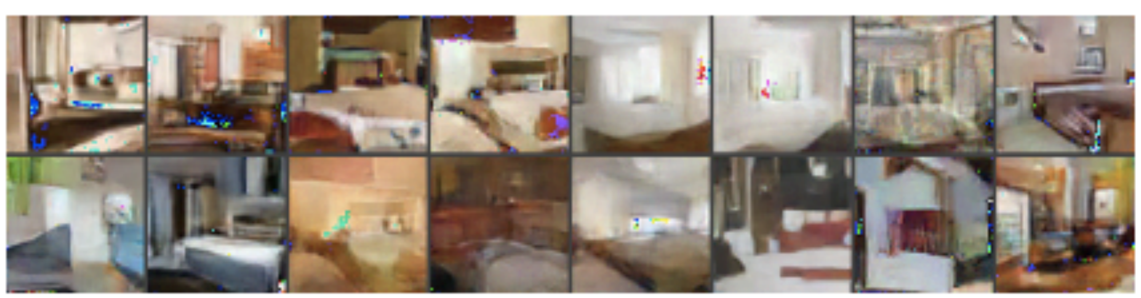
WGAN with DCGAN generator

GAN with DCGAN generator

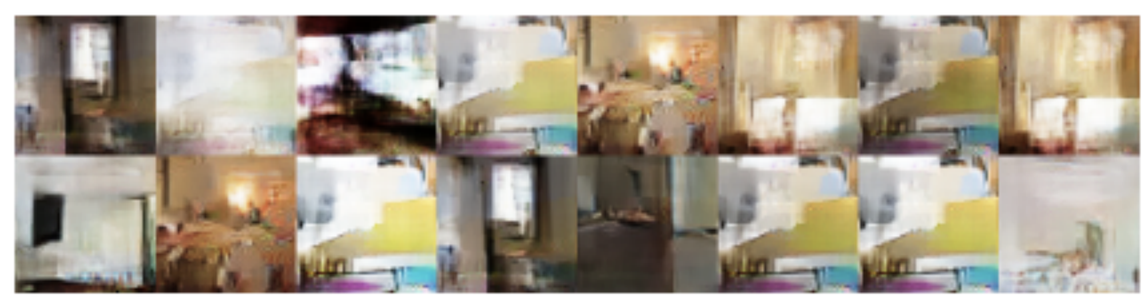
WGAN with DCGAN generator(without BN)



GAN with DCGAN generator(without BN)



WGAN with MLP generator



GAN with MLP generator

WGAN is more robust.

THANKS!