GENERATIVE MODELS

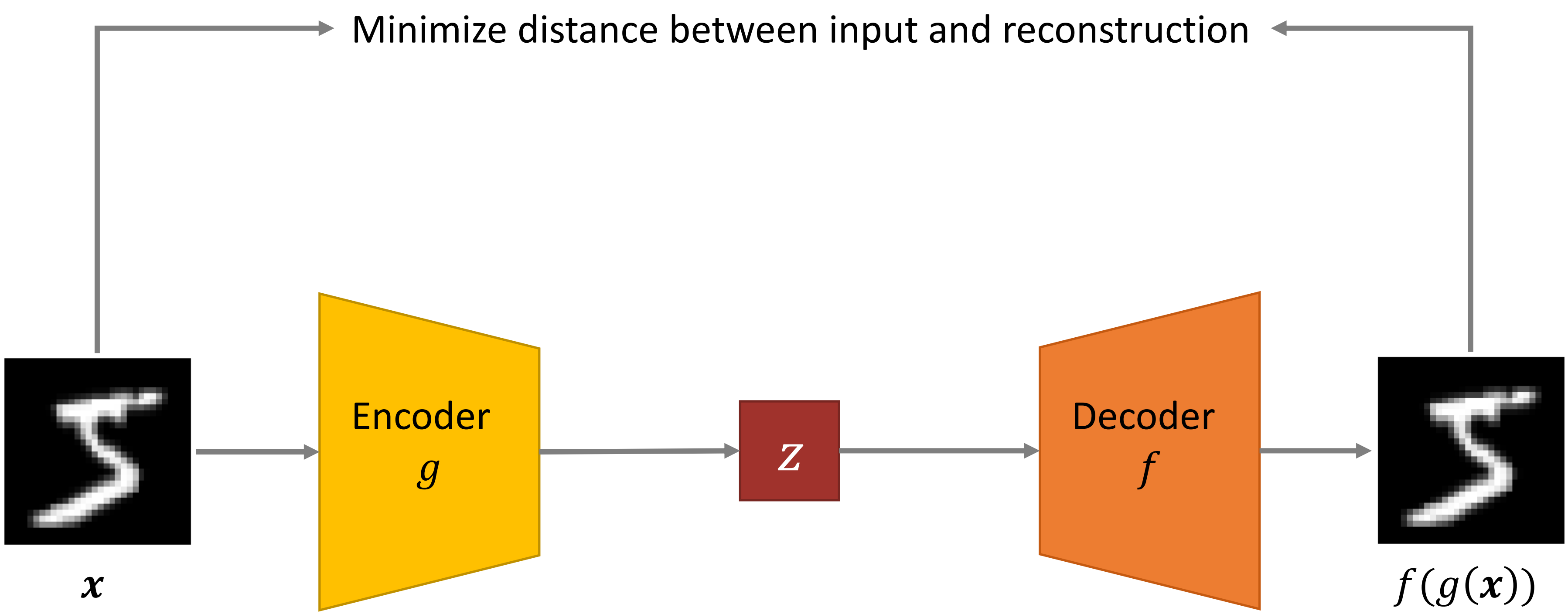
Deep Learning for Computer Vision

Arthur Douillard

https://arthurdouillard.com/deepcourse

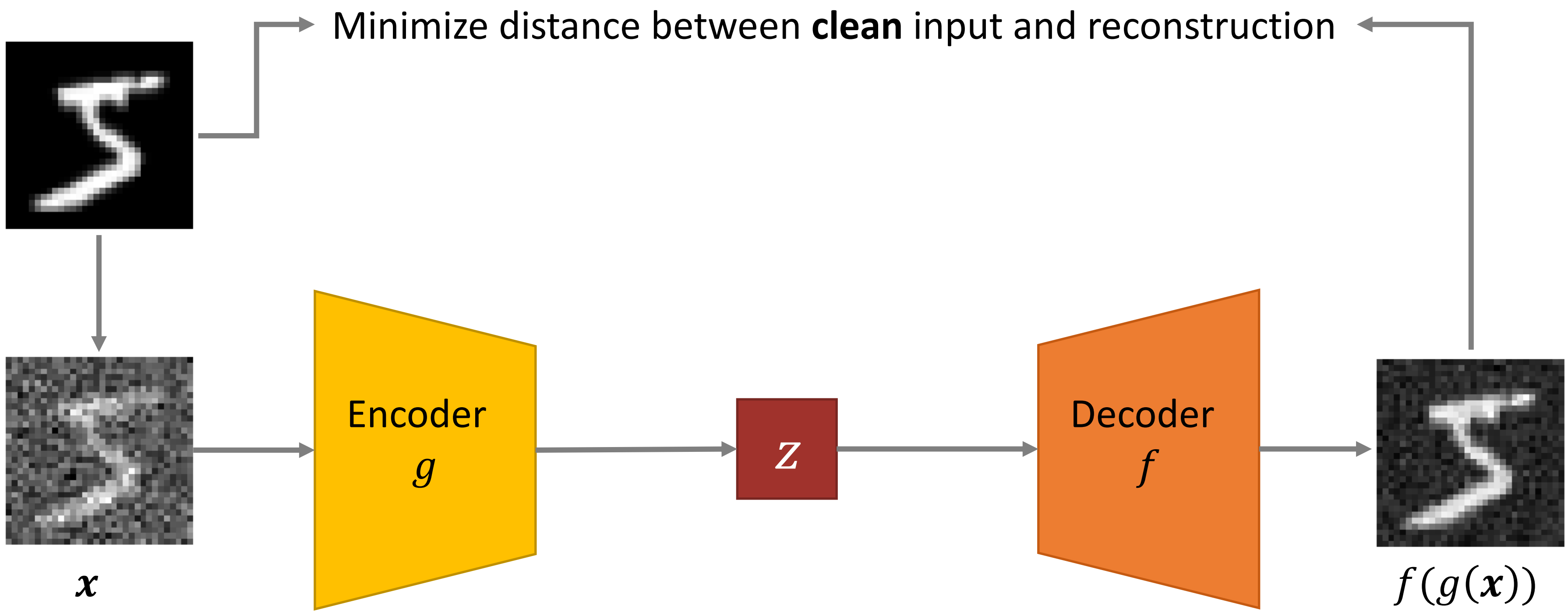
Auto-Encoder

(Under-Complete) Auto-Encoder



- The smaller is the bottleneck z, fewer important features are kept
- Similar to other methods of dimensionality reduction like PCA

Denoising Auto-Encoder



- Add noise to input, try to **denoise it** by reconstruction with clean input

Variational
Auto-Encoder

Variational Auto-Encoder

Instead of producing a deterministic latent code z, can we generate a **distribution**?
→ Generate new images, not simply a reconstruction, but sampling from it

Could we force that the latent code's dimensions are **disentangled**?
→ Modify only an aspect of the image (e.g. keep the face but make hair blond)

[Kingma and Welling, ICLR 2013]

Variational Auto-Encoder

Given the:
- Prior $p(\boldsymbol{z})$
- Likelihood $p(\boldsymbol{x} \mid \boldsymbol{z})$
- Posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$
- Evidence $p(\boldsymbol{x})$

We want to estimate the **posterior**, aka what should be our latent code given $\boldsymbol{x}$**.**

By Bayes and then multiplication rule, we have:

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x,z)}{p(x)}$$

Problem: the evidence $p(\boldsymbol{x})$ is **untractable** (aka it's hard to compute)

Thus, using **variational inference** we are going to approximate the posterior $p(\boldsymbol{z} \mid \boldsymbol{x})$ by a distribution $q(\boldsymbol{z})$ that we defined to be tractable.

Our goal is to minimize the divergence between them:

$$min\, KL(q(z)||p(z|x)) = -\sum q(z) \log \frac{p(z|x)}{q(z)}$$

Variational Auto-Encoder

$$KL(q(z)||p(z|x)) = -\sum q(z)\log\frac{p(z|x)}{q(z)}$$

$$= -\sum q(z)\log\frac{\frac{p(x,z)}{p(x)}}{\frac{q(z)}{1}}$$

$$p(z|x) = \frac{p(x|z)p(z)}{p(x)} = \frac{p(x,z)}{p(x)}$$

Variational Auto-Encoder

$$
\begin{aligned}
KL(q(z)||p(z|x)) &= -\sum q(z)\log\frac{p(z|x)}{q(z)} \\
&= -\sum q(z)\log\frac{\frac{p(x,z)}{p(x)}}{\frac{q(z)}{1}} \\
&= -\sum q(z)\log\frac{p(x,z)}{q(z)}\cdot\frac{1}{p(x)} \\
&= -\sum q(z)[\log\frac{p(x,z)}{q(z)} - \log p(x)] \qquad \text{By the properties of log}
\end{aligned}
$$

Variational Auto-Encoder

$$
\begin{aligned}
KL(q(z)||p(z|x)) &= -\sum q(z) \log \frac{p(z|x)}{q(z)} \\
&= -\sum q(z) \log \frac{\frac{p(x,z)}{p(x)}}{\frac{q(z)}{1}} \\
&= -\sum q(z) \log \frac{p(x,z)}{q(z)} \cdot \frac{1}{p(x)} \\
&= -\sum q(z)[\log \frac{p(x,z)}{q(z)} - \log p(x)] \qquad \text{By the properties of log} \\
&= -\sum q(z) \log \frac{p(x,z)}{q(z)} + \sum q(z) \log p(x) \\
&= -\sum q(z) \log \frac{p(x,z)}{q(z)} + \log p(x) \sum q(z)
\end{aligned}
$$

Because we integrate/sum over $\boldsymbol{z}$ and not $\boldsymbol{x}$

Variational Auto-Encoder

$$
\begin{aligned}
KL(q(z)||p(z|x)) &= -\sum q(z)\log\frac{p(z|x)}{q(z)}\\
&= -\sum q(z)\log\frac{\frac{p(x,z)}{p(x)}}{\frac{q(z)}{1}}\\
&= -\sum q(z)\log\frac{p(x,z)}{q(z)}\cdot\frac{1}{p(x)}\\
&= -\sum q(z)[\log\frac{p(x,z)}{q(z)} - \log p(x)] && \text{By the properties of log}\\
&= -\sum q(z)\log\frac{p(x,z)}{q(z)} + \sum q(z)\log p(x)\\
&= -\sum q(z)\log\frac{p(x,z)}{q(z)} + \log p(x)\sum q(z)\\
&= -\sum q(z)\log\frac{p(x,z)}{q(z)} + \log p(x) && \text{Because a distribution sum to 1}
\end{aligned}
$$

Variational Auto-Encoder

Re-ordering the term of the equation, we have:

$$\log p(x) = KL(q(z)||p(z|x)) + \sum q(z) \log \frac{p(x,z)}{q(z)}$$

x is a constant in our case (we are training on a fixed dataset), the $\log$ of a probability is ≤ 0, and the KL is always ≥ 0. Therefore if we maximize the **Variational Lower Bound** $\mathcal{L} = \sum q(z) \log \frac{p(x,z)}{q(z)}$, we will minimize the KL as intended:

$$\begin{aligned}
\mathcal{L} &= \sum q(z) \log \frac{p(x,z)}{q(z)} \\
&= \sum q(z) \log \frac{p(x|z)p(z)}{q(z)} \\
&= \sum q(z)[\log p(x|z) \log \frac{p(z)}{q(z)}] \\
&= \sum q(z) \log p(x|z) + \sum q(z) \log \frac{p(z)}{q(z)} \\
&= \sum q(z) \log p(x|z) - KL(q(z)||p(z)) \\
&= E_{q(z)} \log p(x|z) - KL(q(z)||p(z))
\end{aligned}$$

Thus our variational lower bound is made of two terms:

$$\mathcal{L} = E_{q(z)} \log p(x|z) - KL(q(z)||p(z))$$

Reconstruction error:

$E_{q(z)} \log p(x|z) \propto E_{q(z)} \log p(x|\hat{x})$ because the decoder is deterministic. If we choose a tractable distribution for $p(\cdot)$ such as the Gaussian distribution, our conditional probability will look like:

$$p(x|\hat{x}) = e^{-|x-\hat{x}|^2}$$

And its $\log$ by:

$$\log p(x|\hat{x}) = -|x - \hat{x}|^2$$

Which is the **Mean Squared Error**, aka can our model reconstruct correctly the input.

KL Divergence:

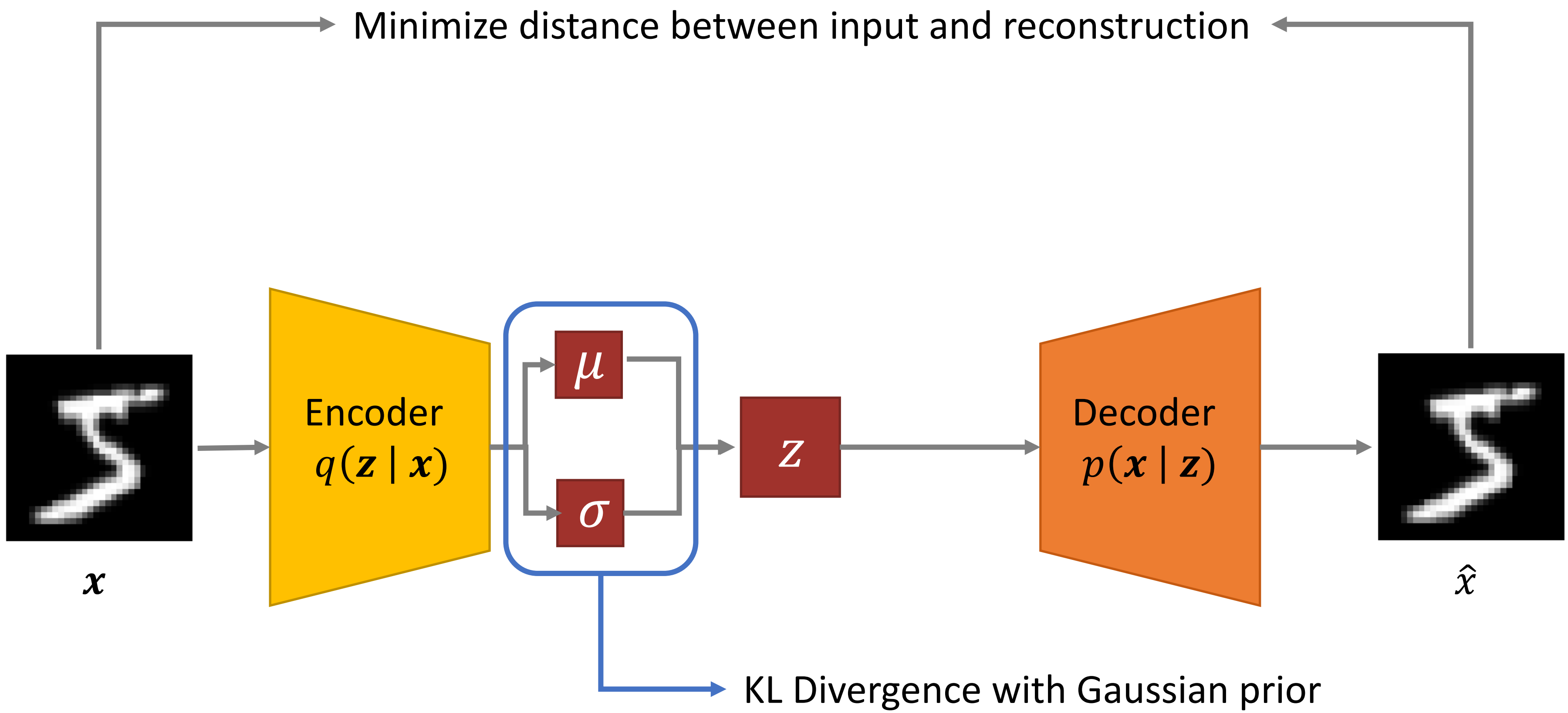
The right part says that our network-distribution $q(z)$ must match the distribution $p(z)$. Now again, we choose $p(\cdot)$ to follow the Gaussian distribution (with zero mean and unit variance $\mathcal{N}(0, I)$).

Now, we are never going to generate z directly by the encoder (it won't be a distribution), but we are going to generate the parameters of the distribution $q(\cdot)$ assuming it's gaussian.

So our KL will be:

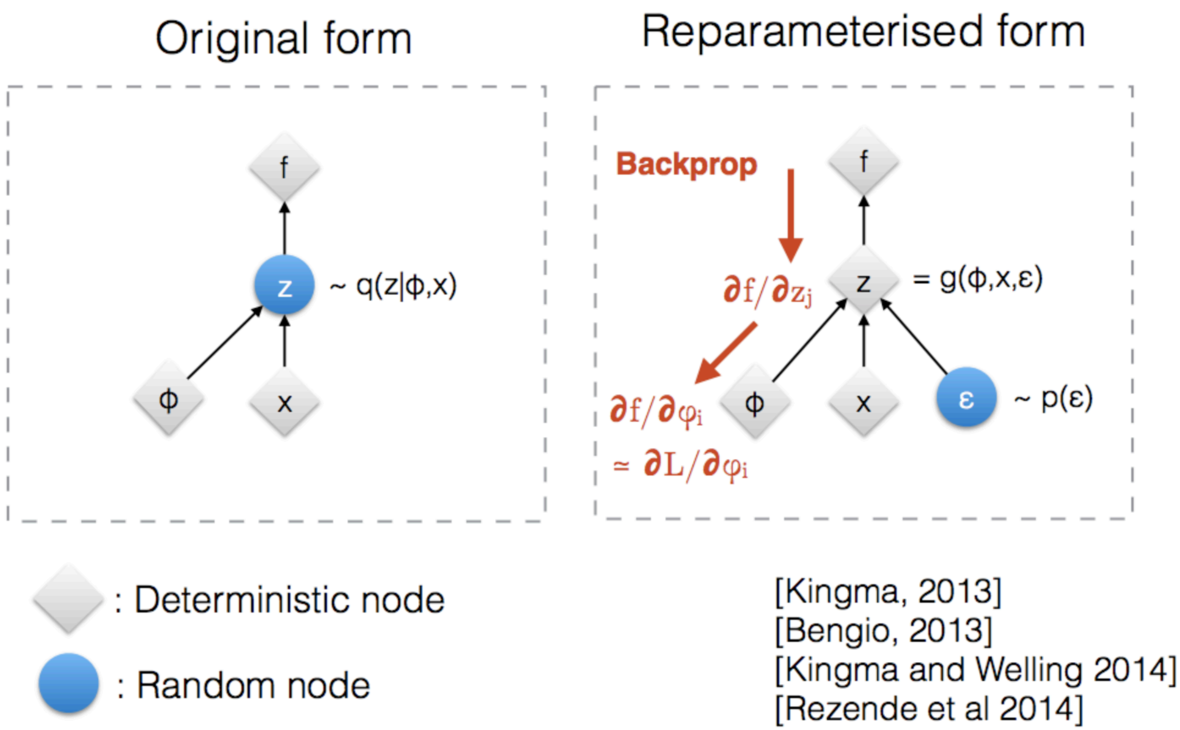
$$KL(\mathcal{N}(\mu, \Sigma)||\mathcal{N}(0, I))$$

Variational Auto-Encoder



The KL Divergence disentangles the latent code by forcing a unique mode per dimension!

Reparametrization Trick

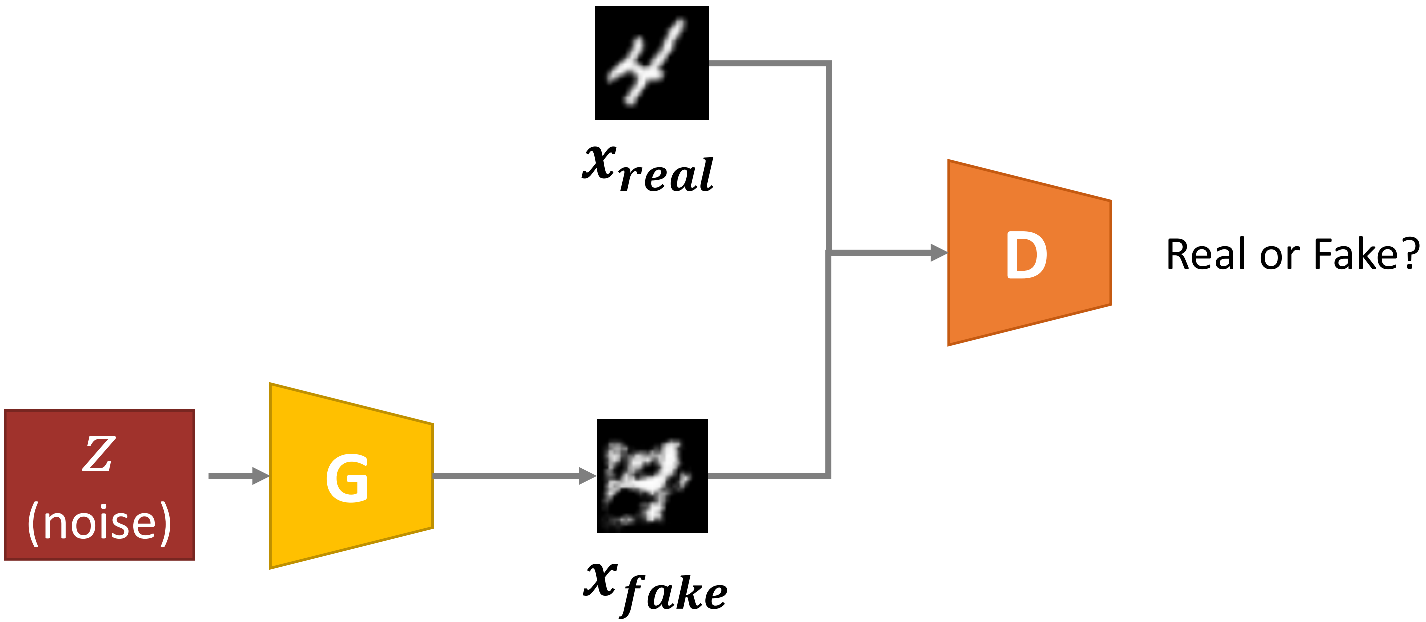


- Sampling operation cannot be backpropagated
- Thus sample a random variable ϵ and multiply the predicted variance σ then add to predicted mean μ

$$\boldsymbol{z} = \mu + \epsilon \times \sigma$$

Generative Adversarial Networks

GAN



Adversarial training between a **discriminator** and a **generator**.

Discriminator has to distinguish between **real and fake images**.

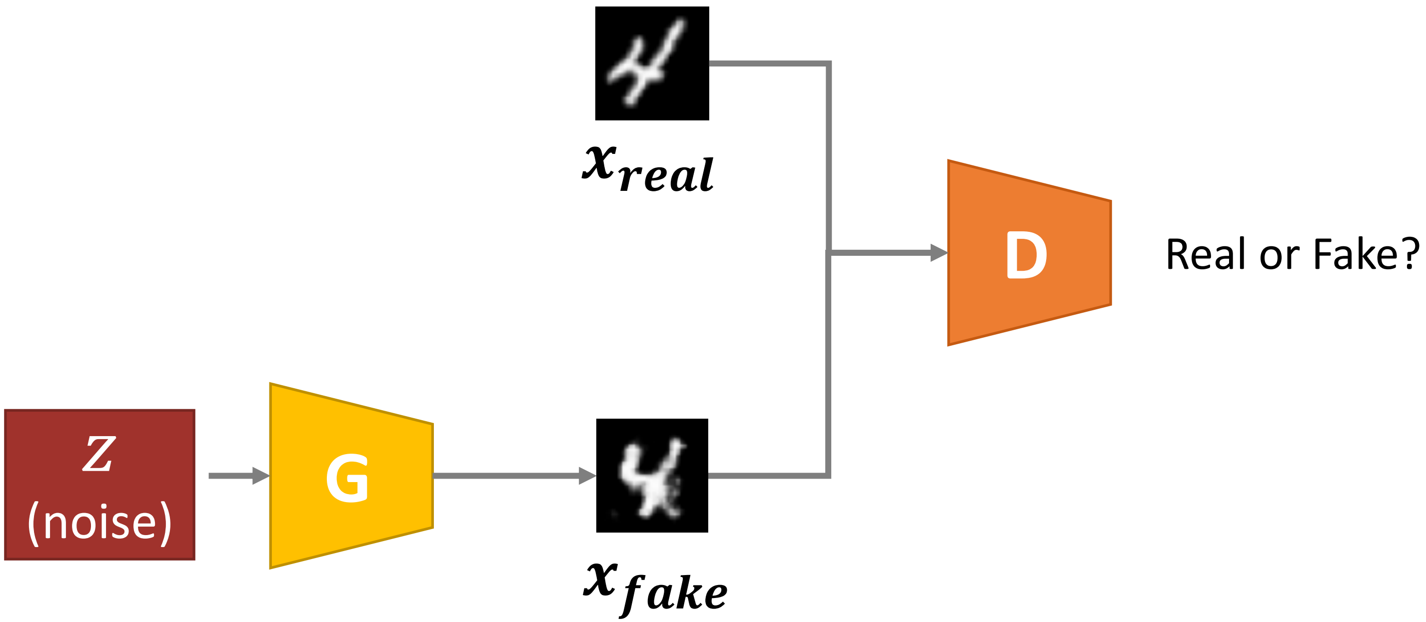
$$\max_D \mathbb{E}_{\mathbf{x}^* \in \mathcal{D}\text{ata}} [\log D(\mathbf{x}^*)] + \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log (1 - D(G(\mathbf{z})))]$$

Generator must fool the discriminator.

$$\max_G \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log D(G(\mathbf{z}))]$$

[Goodfellow et al. NeurIPS 2014]

GAN



Adversarial training between a **discriminator** and a **generator**.

Discriminator has to distinguish between **real and fake images**.

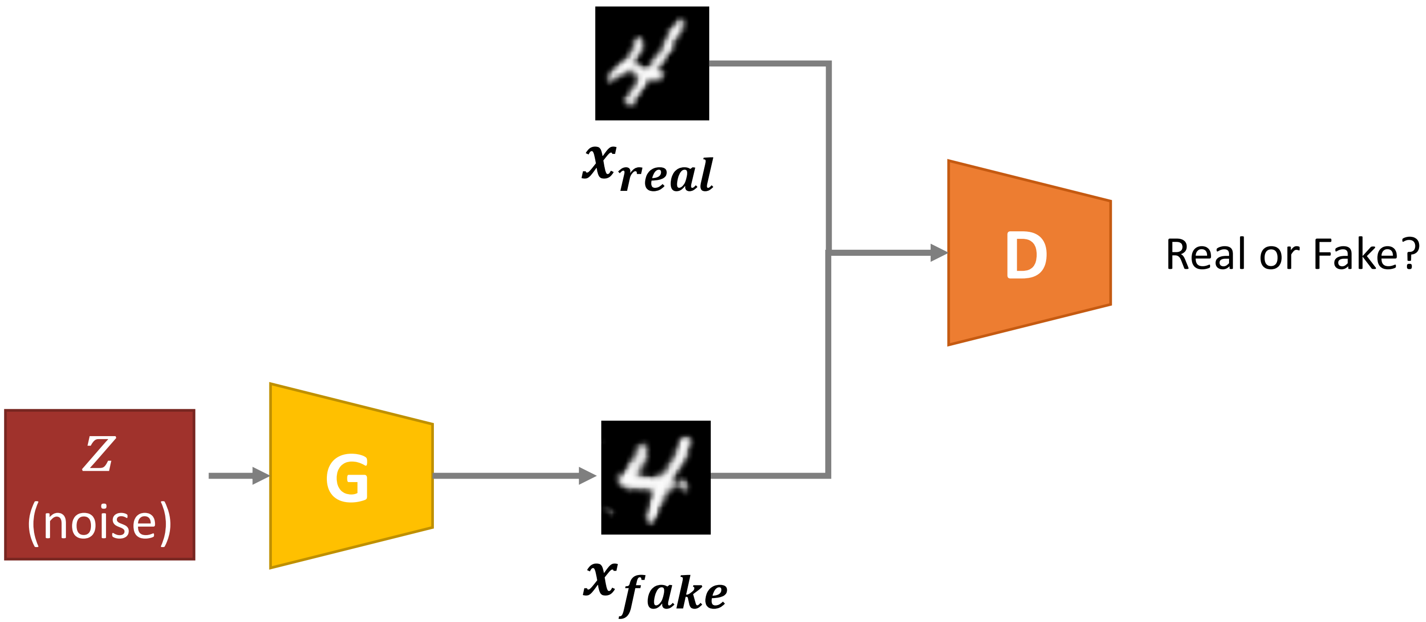
$$\max_D \mathbb{E}_{\mathbf{x}^* \in \mathcal{D}\text{ata}} [\log D(\mathbf{x}^*)] + \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log (1 - D(G(\mathbf{z})))]$$

Generator must fool the discriminator.

$$\max_G \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log D(G(\mathbf{z}))]$$

[Goodfellow et al. NeurIPS 2014]

GAN



Adversarial training between a **discriminator** and a **generator**.

Discriminator has to distinguish between **real and fake images**.

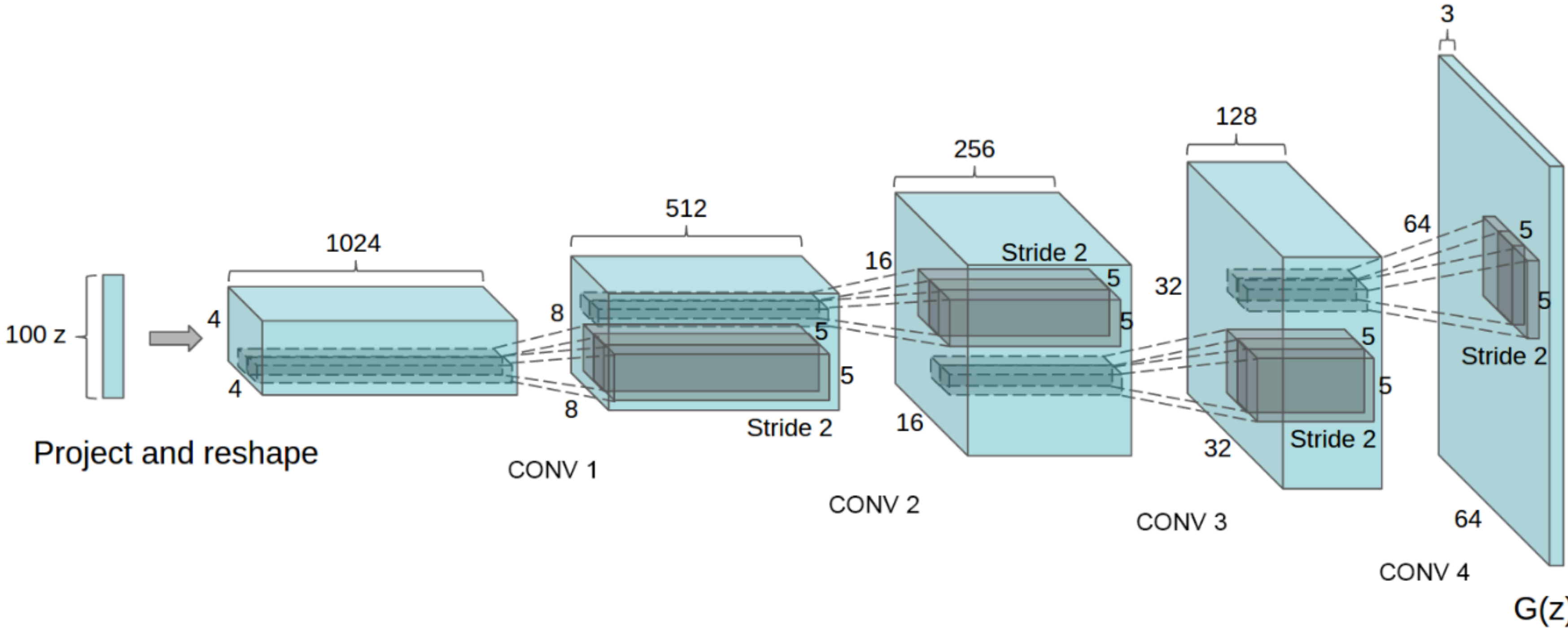
$$\max_D \mathbb{E}_{\mathbf{x}^* \in \mathcal{D}\text{ata}} [\log D(\mathbf{x}^*)] + \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log (1 - D(G(\mathbf{z})))]$$

Generator must fool the discriminator.

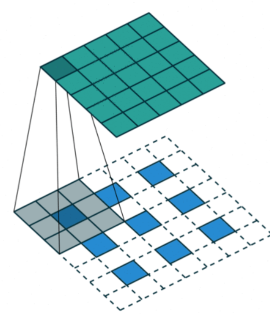
$$\max_G \mathbb{E}_{\mathbf{z} \sim P(\mathbf{z})} [\log D(G(\mathbf{z}))]$$

[Goodfellow et al. NeurIPS 2014]

DC-GAN

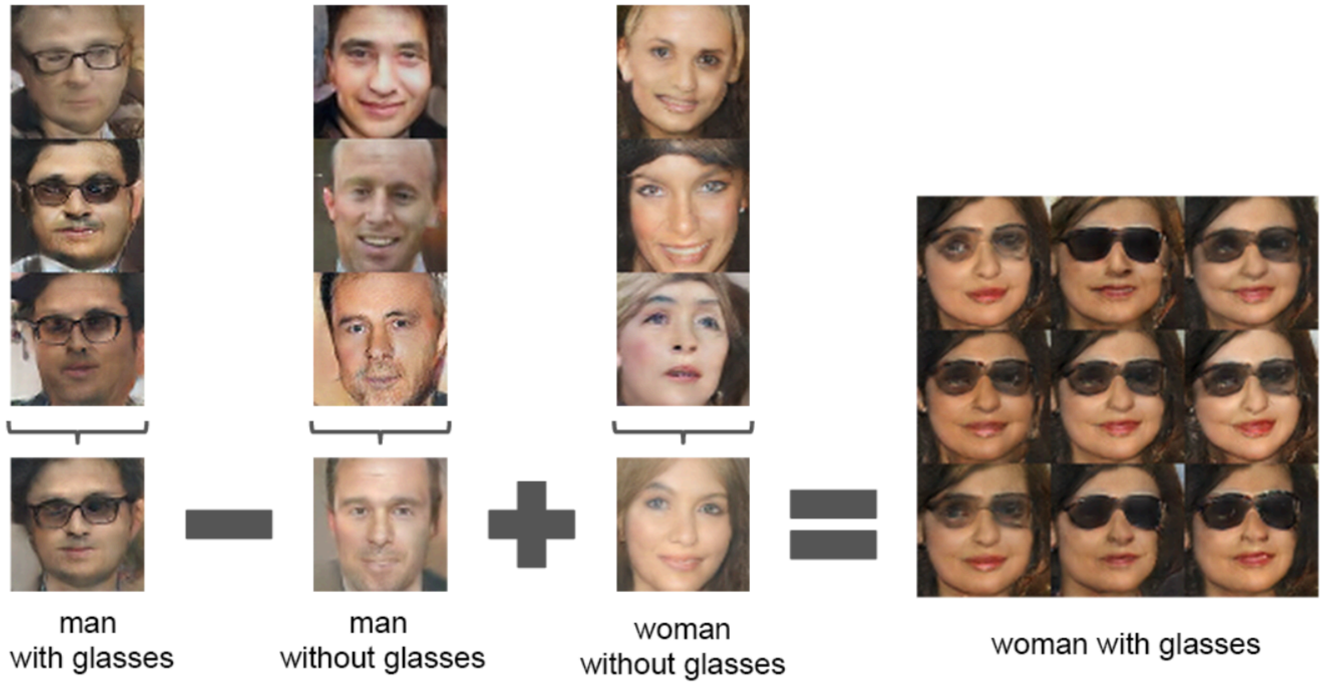


- Use convolutions instead of FC layers
- Upsample using **transposed convolutions**



[Radford et al. ICLR 2016]

DC-GAN

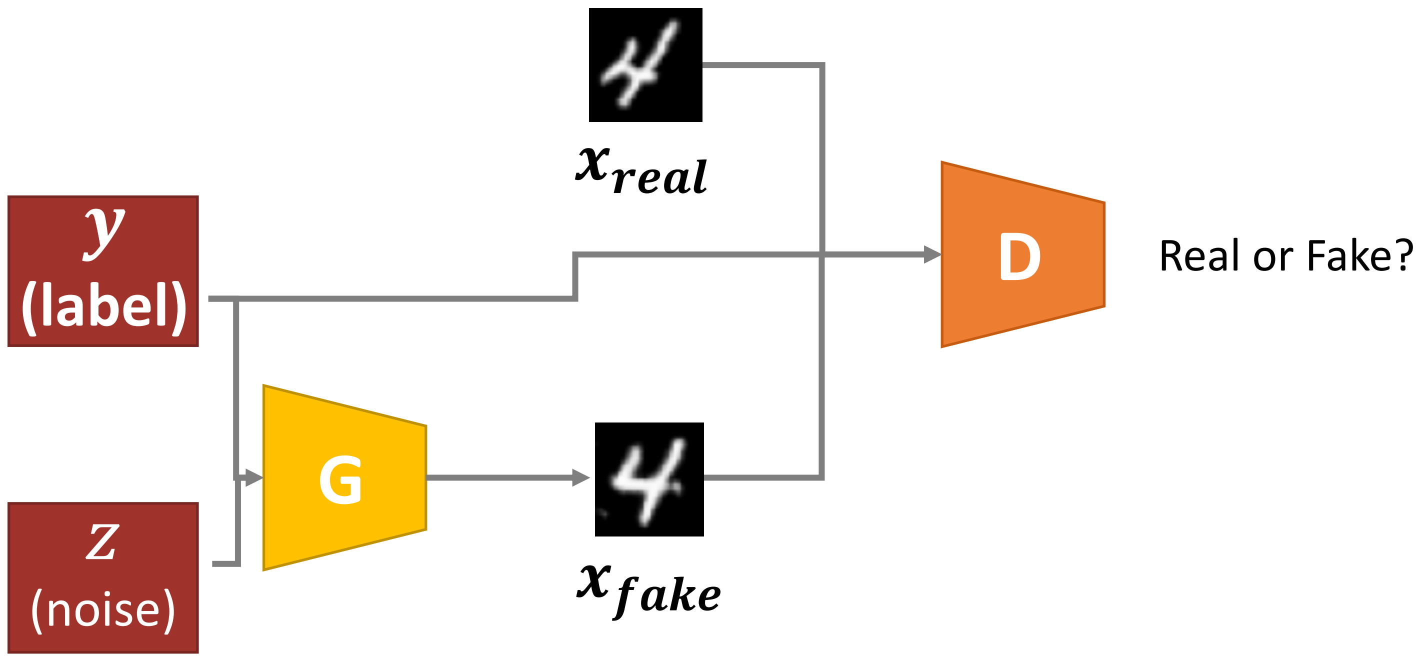


Vector arithmetic for visual concepts:

1. Find many noise vectors that produce man with glasses, man w/o glasses, etc.
2. Average noise vectors per category
3. Do some basic arithmetic with the noise vectors
4. Generate!

[Radford et al. ICLR 2016]

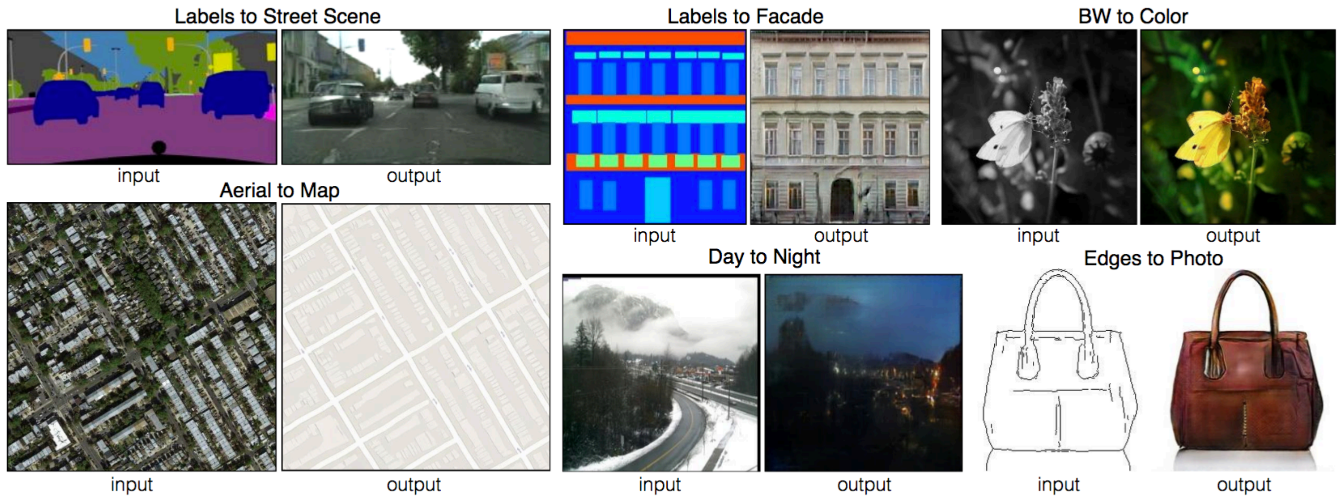
cGAN: Conditional-GAN

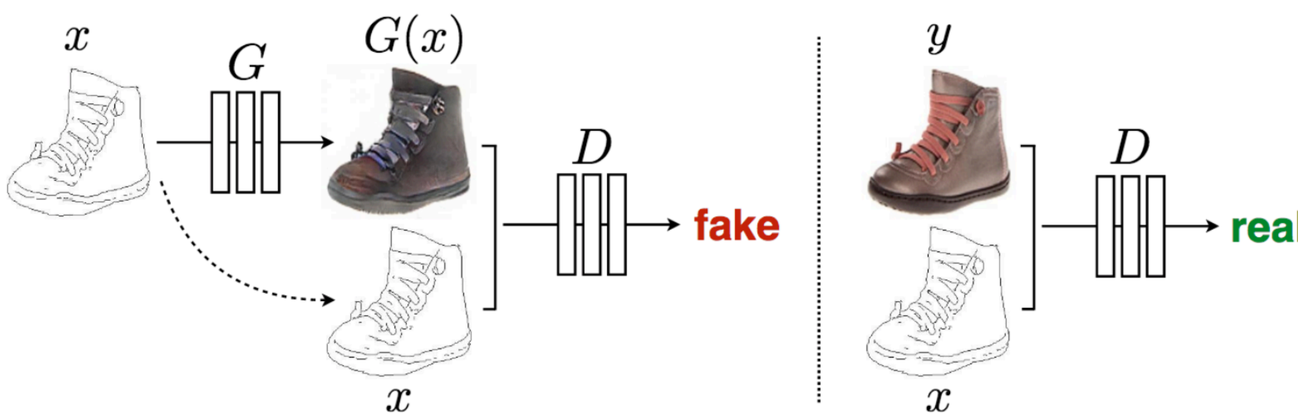


- Add label in input to both the generator and discriminator
- Now the generator, given label "4" will not tolerate a "5" even if it's very realistic

[Mirza and Osindero, NeurIPS 2014]

Pix2Pix

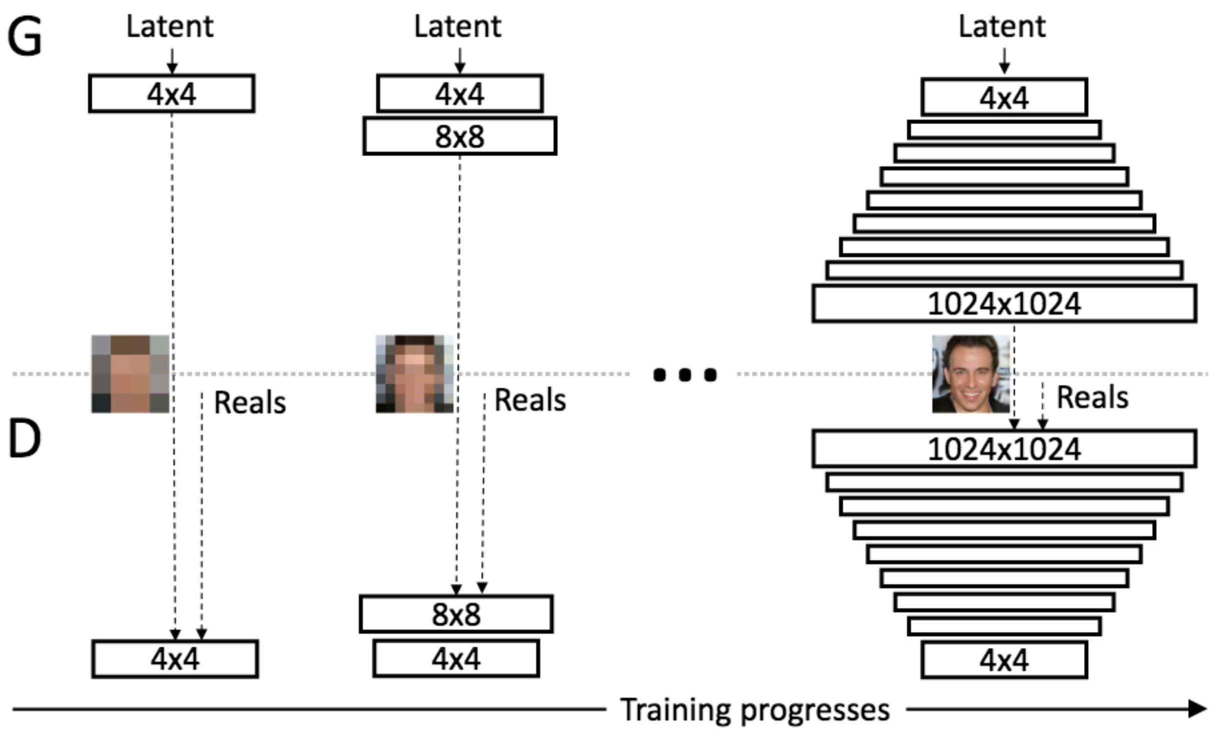




- Like cGAN but conditioned with various kind of data (segmentation, maps, drawing, etc.)

[Isola et al. CVPR 2017]

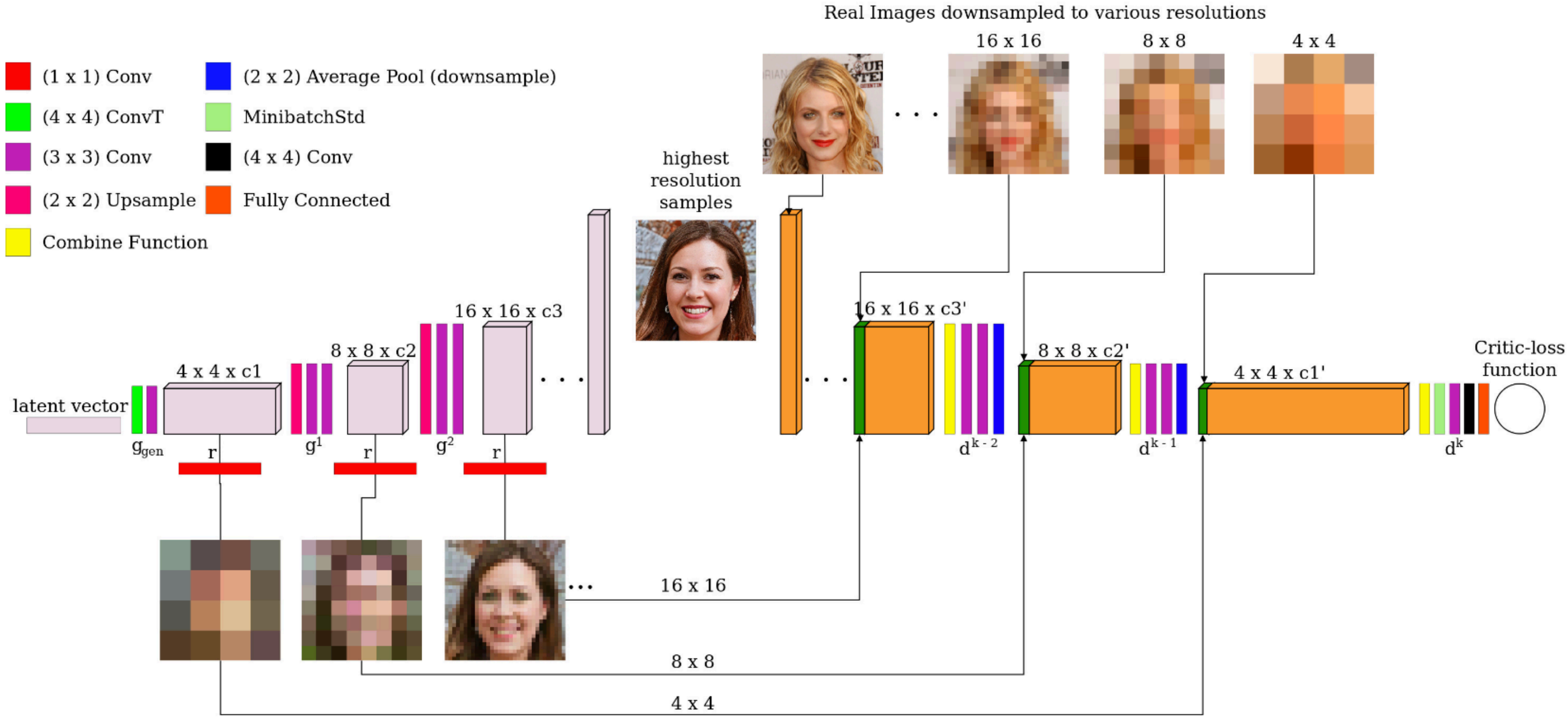
ProGAN: Progressive growing



- Generate progressively higher resolution images by extending the architecture
- Akin to curriculum learning

[Karras et al. ICLR 2018]

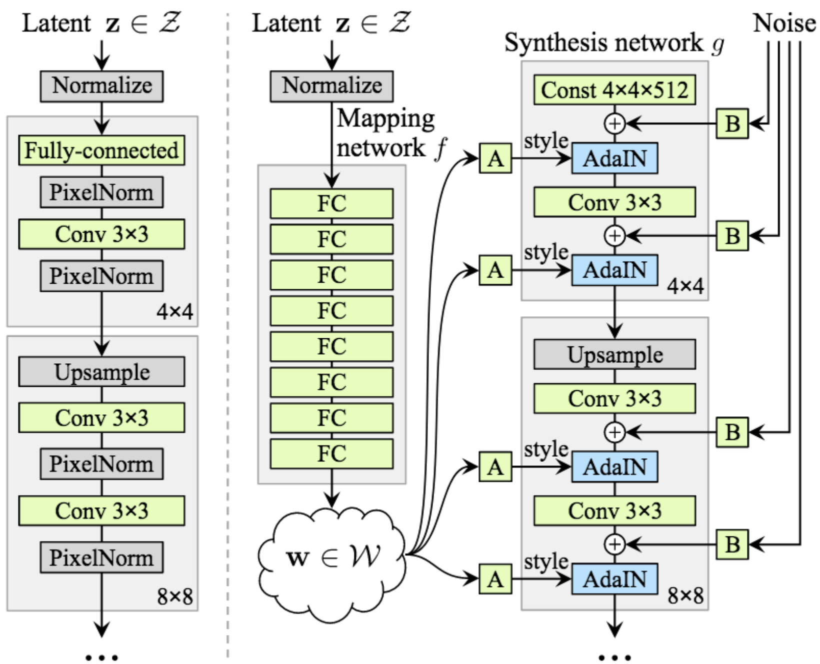
MSG-GAN: Multi-Scale Gradients GAN



- Synthetize in the same time all resolutions
- Simpler architecture than ProGAN and much faster to converge with better results

[Karnewar and Wang, CVPR 2020]

StyleGAN



$$\mathrm{AdaIN}(x, y) = \sigma(y)\left(\frac{x - \mu(x)}{\sigma(x)}\right) + \mu(y)$$

- Based on ProGAN
- Mapping network transforms latent vector noise z
- Which is then added at multiple level with AdaIN
- Latent vector is more disentangled leading to easier vector arithmetic because of the separation of style and stochastic variations

[Karras et al. CVPR 2019]

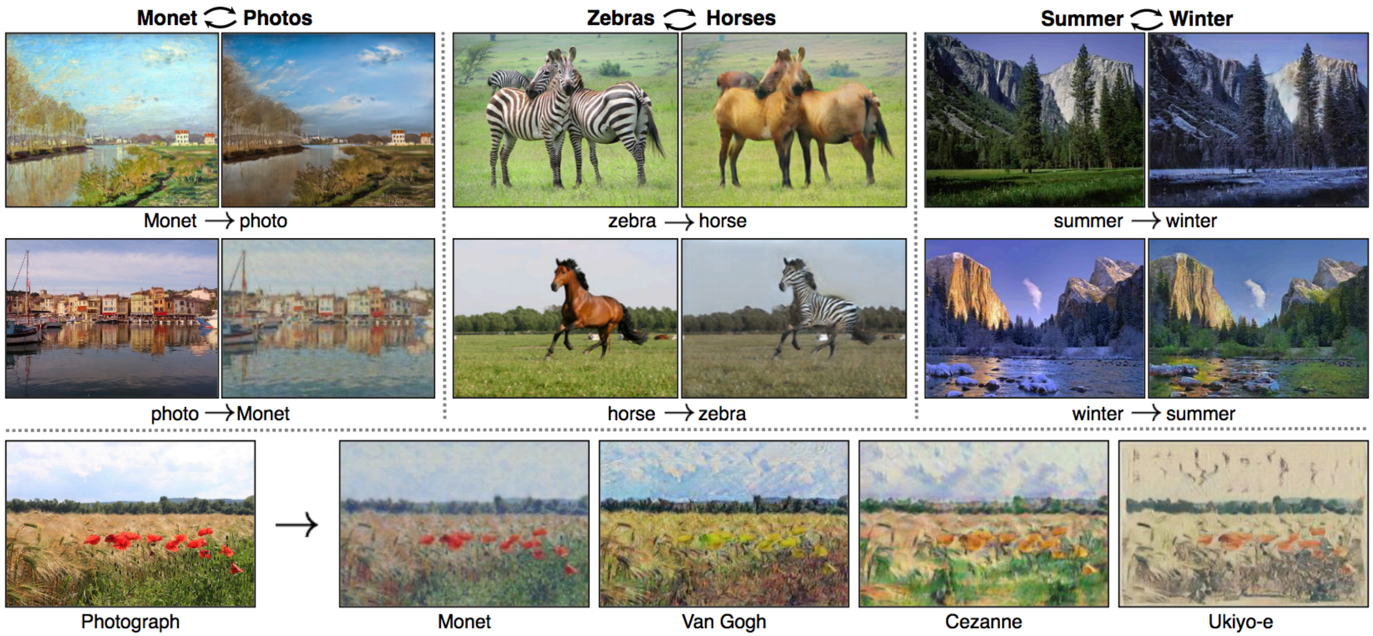
Based on StyleGAN

Which face is real?

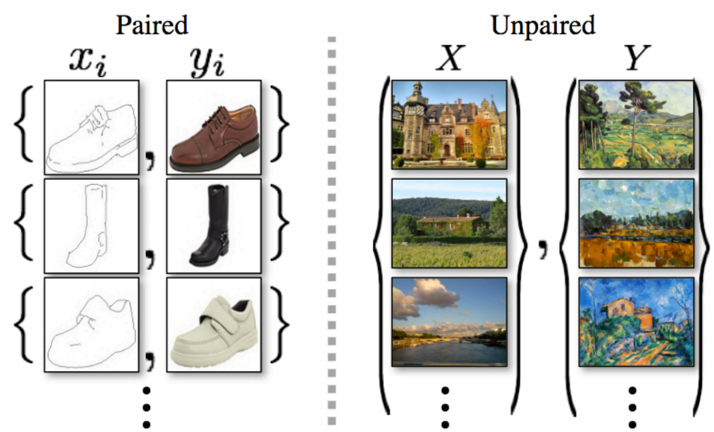
whichfaceisreal.com

[[Karras et al. CVPR 2019]]

CycleGAN

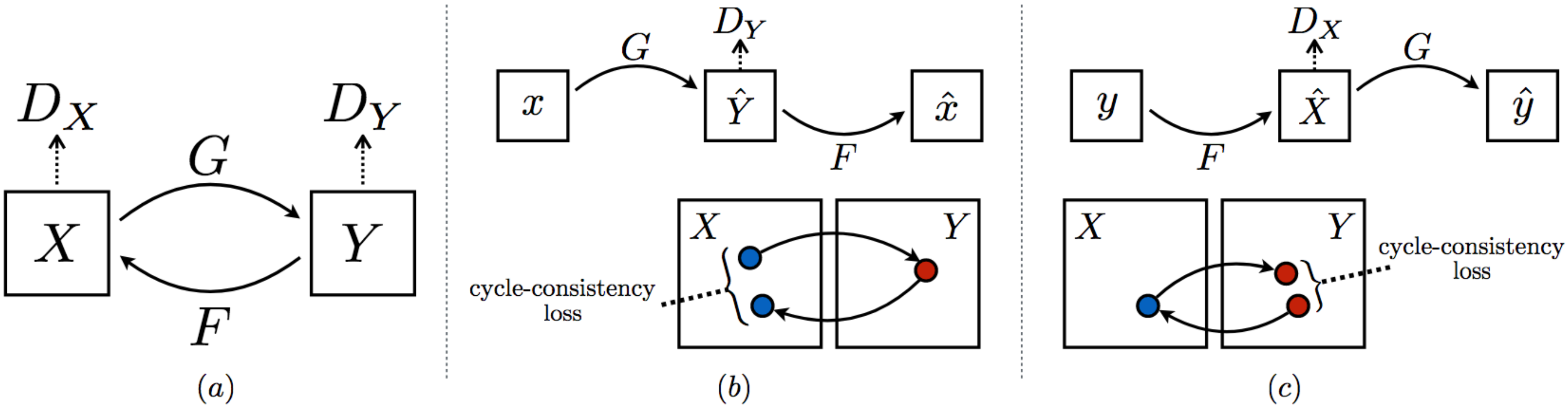


- Unpaired Image-to-Image Translation
- No need to have matching domains!

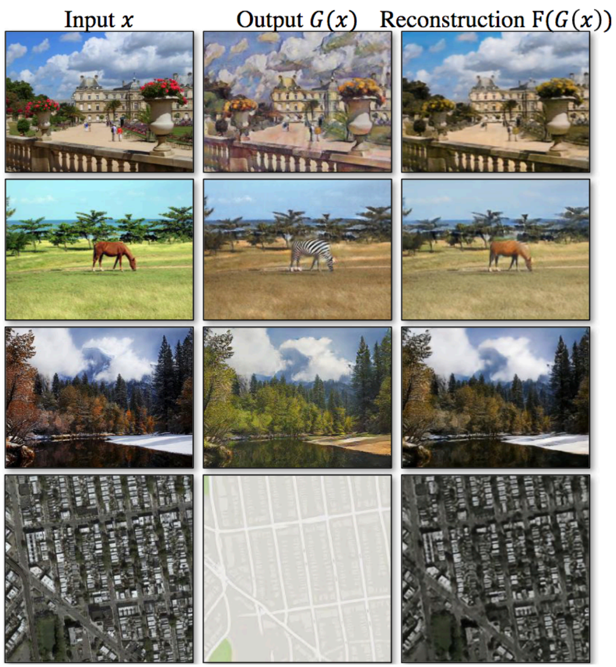


[Zhu et al. ICCV 2017]

CycleGAN



- Translate image from domain X to Y then back to X
  - And vice-versa
- When in domain Y, a discriminator determines if domain is correct



[Zhu et al. ICCV 2017]

Is My Generative Model Good?

IS: Inception Score

$$\mathrm{IS}(G) = \exp\left(\mathbb{E}_{\mathbf{x}\sim p_g} D_{KL}\left(p(y|\mathbf{x}) \parallel p(y) \right) \right)$$

1. Produce likelihoods p(y|x) with a pre-trained Inception network
2. Average likelihoods to have marginal probability p(y)
3. Compute KL divergence between them + average over multiple split + exp

Higher is better, minimum score is 0.

$$p(y) = \int_{\mathbf{x}} p(y|\mathbf{x}) p_g(\mathbf{x})$$

We want:

- A low-entropy conditional probability p(x|y) (aka high confidence on a class label)
- A high-entropy marginal probability p(y) to have more diversity

https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

FID: Frélet Inception Distance

$$\mathrm{FID} = |\mu - \mu_w|^2 + \mathrm{tr}(\Sigma + \Sigma_w - 2(\Sigma\Sigma_w)^{1/2})$$

1. Extract features at a deep but not last layer for both real and generated images
2. Minimize this distance between on the mean and covariance activations

Lower is better, minimum score is 0.

Human

Amazon Mechanical Turk

Access a global, on-demand, 24x7 workforce

Get started with Amazon Mechanical Turk

Amazon Mechanical Turk (MTurk) is a crowdsourcing marketplace that makes it easier for individuals and businesses to outsource their processes and jobs to a distributed workforce who can perform these tasks virtually. This could include anything from conducting simple data validation and research to more subjective tasks like survey participation, content moderation, and more. MTurk enables companies to harness the collective intelligence, skills, and insights from a global workforce to streamline business processes, augment data collection and analysis, and accelerate machine learning development.

- Often papers also use humans to validate if a model X produces images more realistic than a model Y
- Platform like Mechanical Turk can do that, but there is a cost!

Style Transfer
Auto-Regressive
Normalizing Flows
NeRF

Style Transfer

Given pretrained VGG16 network:

1. Clone content image as X
2. Minimize **content loss** (MSE) between X and Content image
   Minimize **style loss** (MSE) between gram matrices of X and Style image

All that is done at the features level not at the pixels level.

[Gatys et al. CVPR 2016]

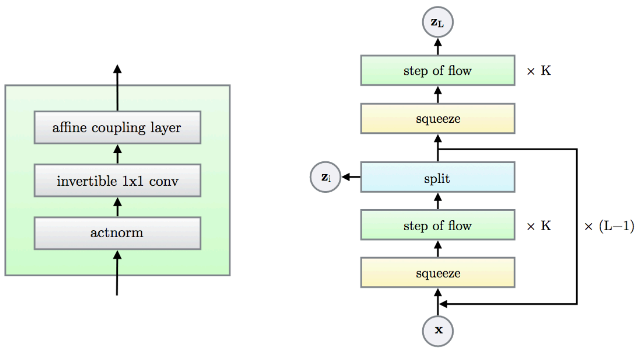
Auto-Regressive

$$p_\theta(\mathbf{x}) = \prod_{i=1}^{N} p_\theta(x_i|\mathbf{x}_{<i})$$

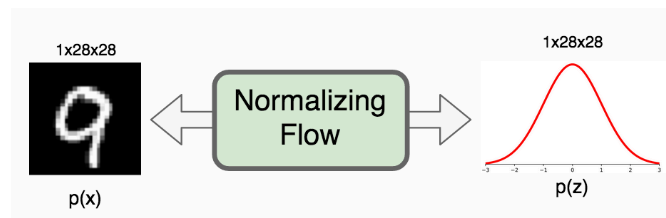
- Image generation as a sequence modeling task, akin to language model in NLP
- Can be slow to generate because of a limited parallelizability
- Can suffer from very long sequences

[PixelRNN, van den Oord et al. ICLR 2016], [Sparse Transformer, Child et al. arXiv 2019]

Normalizing Flows



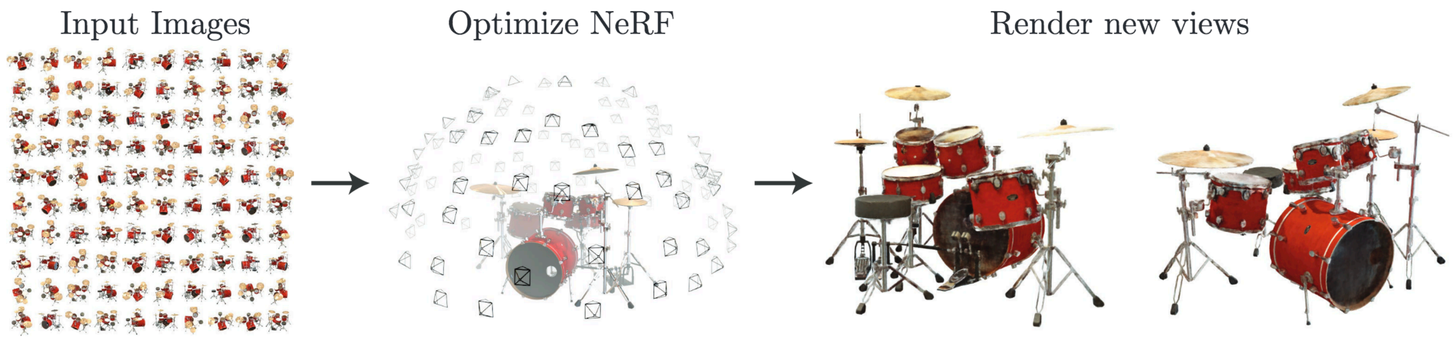
(a) One step of our flow. (b) Multi-scale architecture (Dinh et al., 2016).

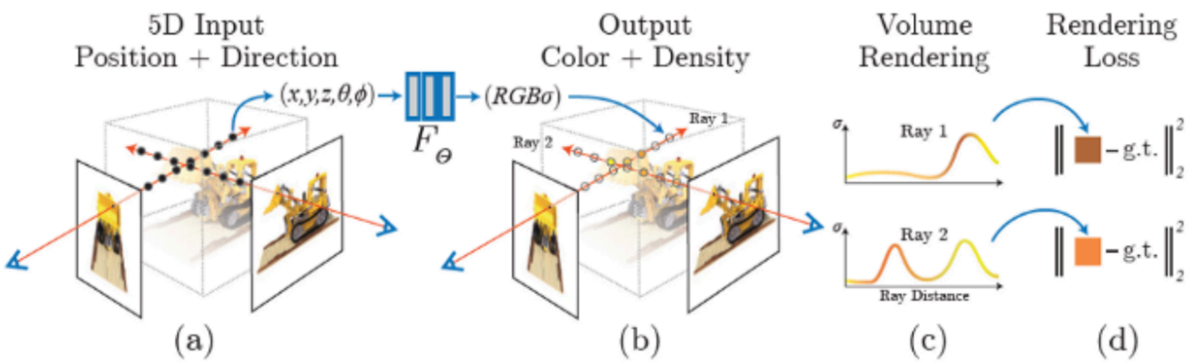


- Whole network is invertible (thus no pooling, need to have same dimension through the network!)
- During training learn the mapping $f(x) = z$, with z a multivariate Gaussian
- During inference, generate images by sampling z and doing the inverse mapping $f^{-1}(z)$

[Glow, Kingma and Dhariwal, NeurIPS 2018], third image source

NeRF: Neural Radiance Fields





- Volume rendering similar to ray tracing, learn to generate a pixel color and a volume density given a 3D location and 2D viewing direction
- Can be used extrapolate new views
- Use 9 fully connected layers + ReLU

[NeRF, Mildenhall et al. ECCV 2020]

Small break,
then coding session!