Deep Learning for Computer Vision

Fall 2022

https://cool.ntu.edu.tw/courses/189345 (NTU COOL)

http://vllab.ee.ntu.edu.tw/dlcv.html (Public website)

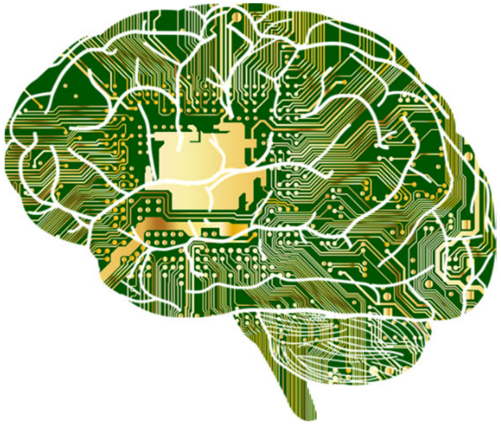
Yu-Chiang Frank Wang 王鈺強, Professor

Dept. Electrical Engineering, National Taiwan University

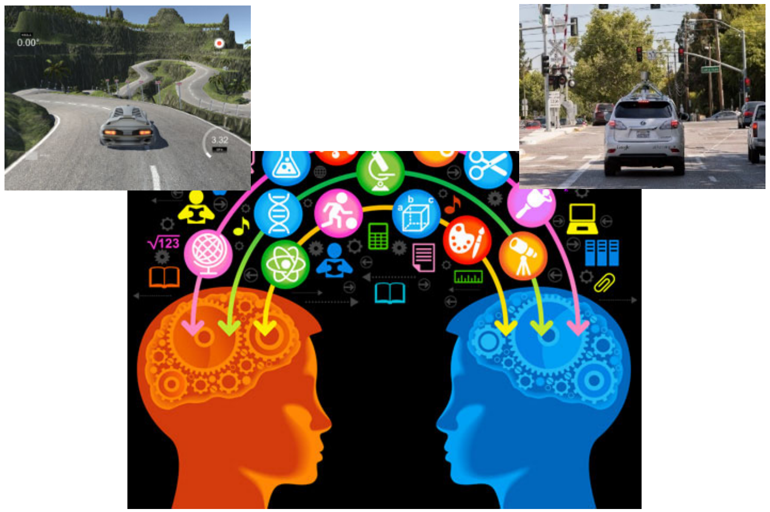
2022/10/11

What to Be Covered Today…

- Generative Models
  - Diffusion Model
- Transfer Learning
  - Visual Classification – Domain Adaptation
  - Visual Synthesis – Style Transfer
- Representation Disentanglement
  - Supervised vs. unsupervised feature disentanglement

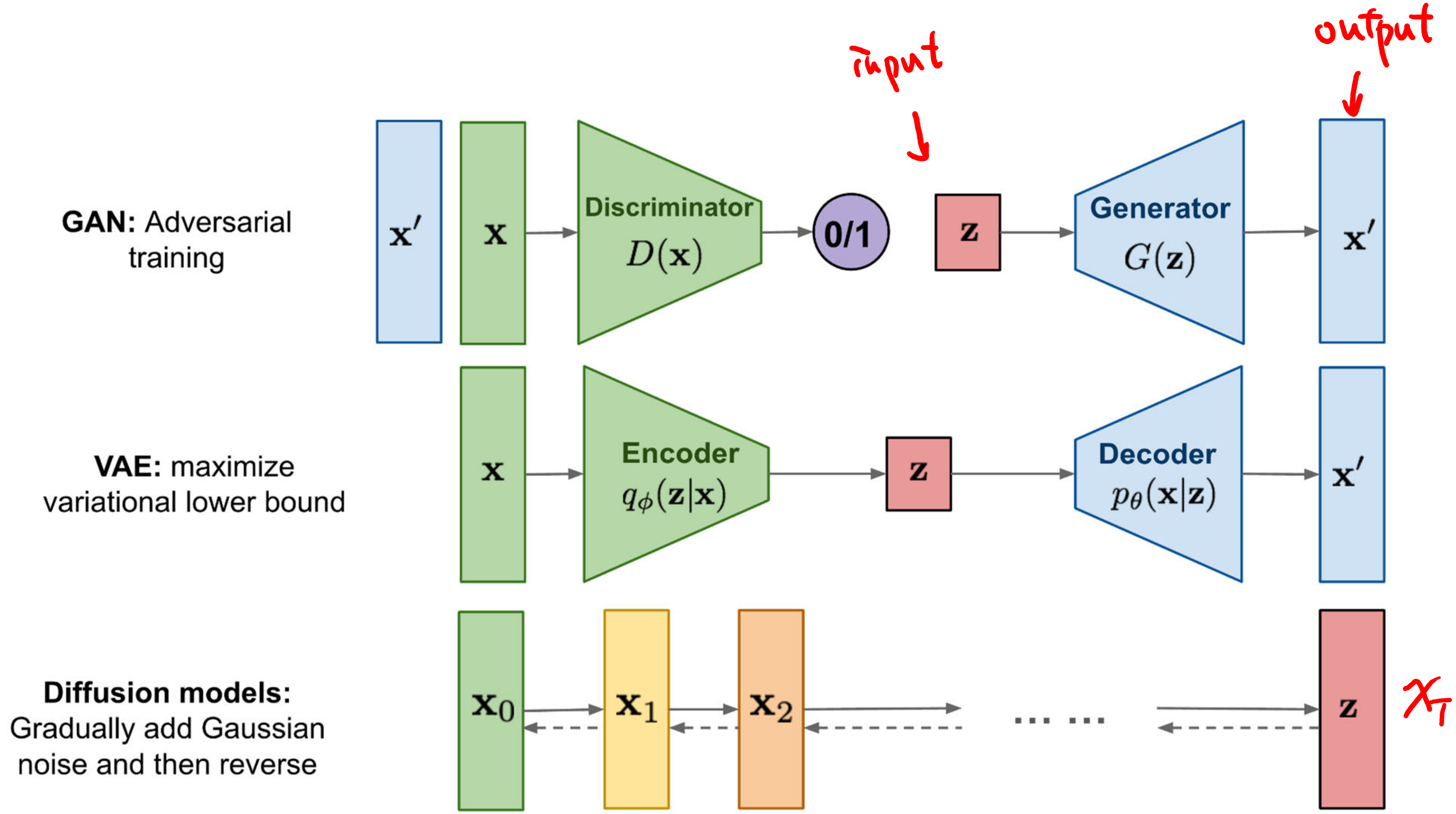


Sprouts in the shape of text 'Imagen' coming out of a fairytale book.



From VAE to Diffusion Model

$p(x)$: data distribution



Denoising Diffusion Models

- Emerging as powerful generative models
  - Unconditional image synthesis
  - Conditional image synthesis
  - Outperforms GANs

Diffusion Models Beat GANs on Image Synthesis, Dhariwai & Nochol, OpenAI, 2021

Cascaded Diffusion Models for High Fidelity Image Generation, Ho et al., Google, 2021

Slide credit: Kreis, Gao, & Vahdat

Denoising Diffusion Models

- Emerging as powerful generative models
  - Unconditional image synthesis
  - Conditional image synthesis
  - Outperforms GANs

DALL·E 2

text (condition)

"a teddy bear on a skateboard in times square"

Diffusion Models Beat GANs on Image Synthesis, Dhariwai & Nochol, OpenAI, 2021

Imagen

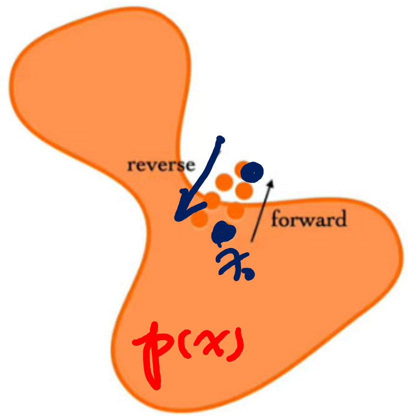
A group of teddy bears in suit in a corporate office celebrating the birthday of their friend. There is a pizza cake on the desk.

Cascaded Diffusion Models for High Fidelity Image Generation, Ho et al., Google, 2021

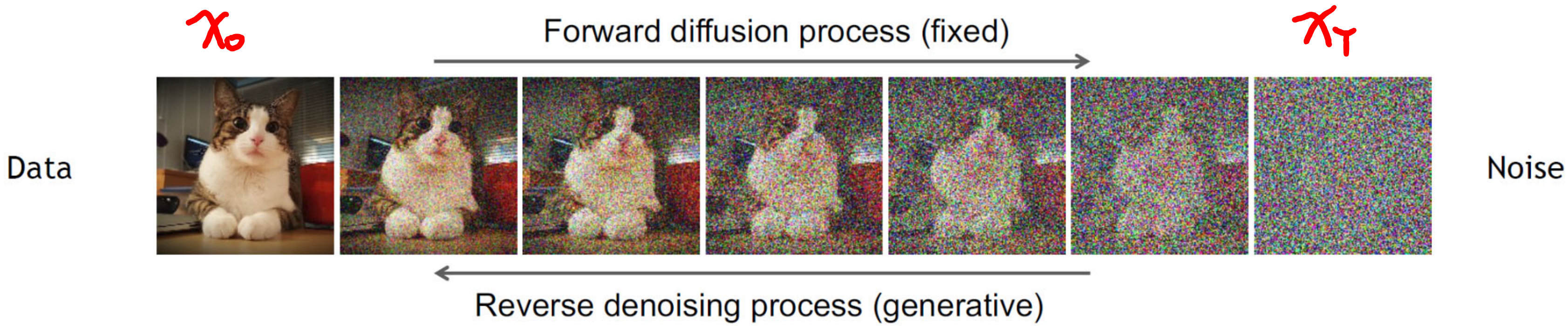
Slide credit: Kreis, Gao, & Vahdat

Denoising Diffusion Models:

Learning to generate by denoising



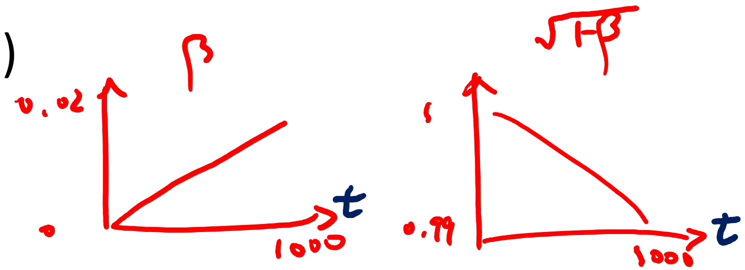
- 2 processes required for training:
  - Forward diffusion process – gradually add noise to input
  - **Reverse diffusion process** – learns to generate/restore data by denoising (typically implemented via a U-net)
  - Comments about noise scheduling (see next slide)



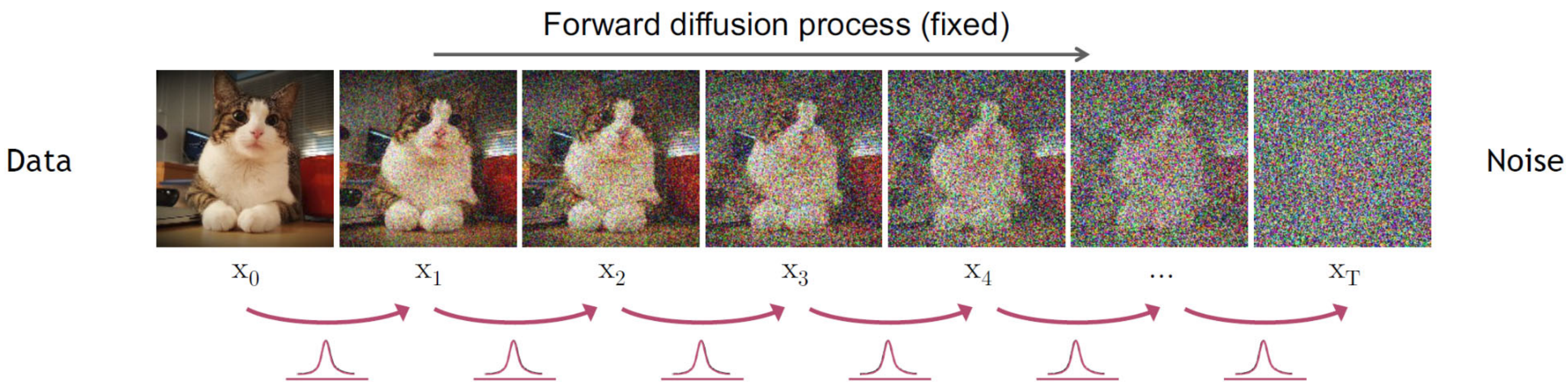
Ho et al., Denoising Diffusion Probabilistic Models, NeurIPS 2020
Song et al., Score-Based Generative Modeling through Stochastic Differential Equations, ICLR 2021

Slide credit: Kreis, Gao, & Vahdat

Denoising Diffusion Models:
Learning to generate by denoising (cont'd)



- Forward diffusion process
  - Gradually add noise to the input in T steps
  - Recall that x_0 denotes clean input image, and x_T is the final noisy one.
  - Comments on $q(x_t|x_{t-1})$



Normal distribution — mean — var

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x_t}; \sqrt{1-\beta_t}\mathbf{x_{t-1}}, \beta_t\mathbf{I}) \Rightarrow q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad \text{(joint)}$$

output

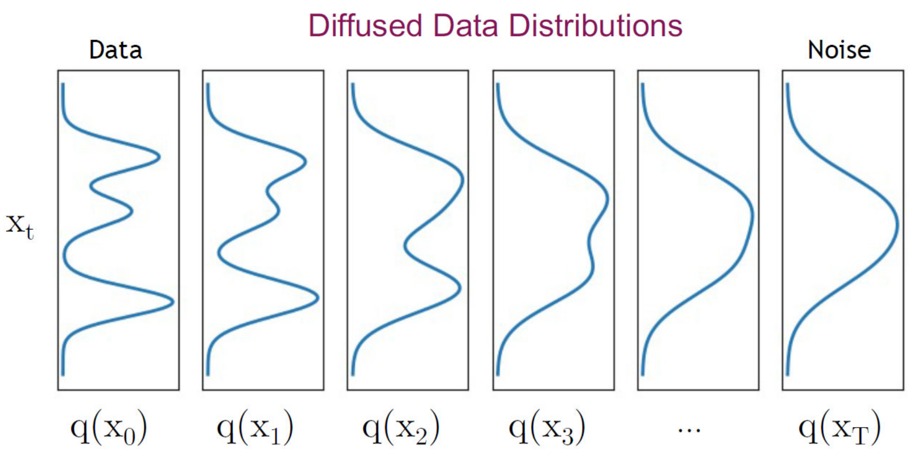
variance schedule (hyperparameter)

Slide credit: Kreis, Gao, & Vahdat

Denoising Diffusion Models:

Learning to generate by denoising (cont'd)

- Forward diffusion process
  - Gradually add noise to the input in T steps (cont'd)
  - Diffusion kernel
  - So what happens to data distribution during this process?



$$\underbrace{q(\mathbf{x}_t)}_{\text{Diffused data dist.}} = \int \underbrace{q(\mathbf{x}_0, \mathbf{x}_t)}_{\text{Joint dist.}} \, d\mathbf{x}_0 = \int \underbrace{q(\mathbf{x}_0)}_{\text{Input data dist.}} \, \underbrace{q(\mathbf{x}_t|\mathbf{x}_0)}_{\text{Diffusion kernel}} \, d\mathbf{x}_0$$

The diffusion kernel is Gaussian convolution.

#7

Denoising Diffusion Models:

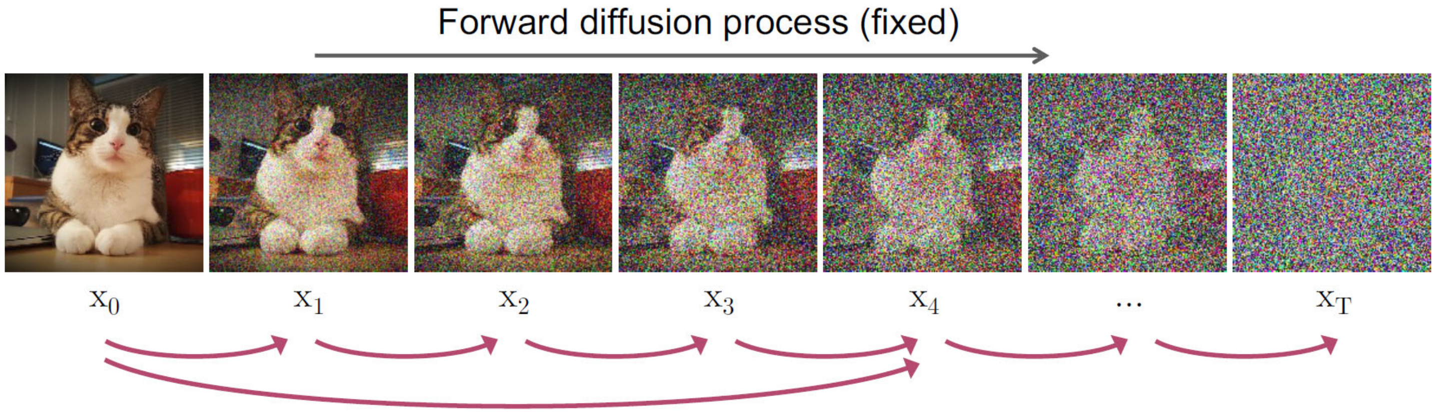
Learning to generate by denoising (cont'd)

repara. trick

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$$
$$= \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}\varepsilon$$
$$= \sqrt{\alpha_t}x_{t-1} + \sqrt{1-\alpha_t}\varepsilon$$
$$= \sqrt{\alpha_t\alpha_{t-1}}x_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\varepsilon$$
$$= \dots$$

- Forward diffusion process
  - Gradually add noise to the input in T steps
  - Diffusion kernel:

Data



Noise

Define $\begin{cases} \alpha_t = 1 - \beta_t \\ \bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s) = \prod_{s=1}^{t}\alpha_s \end{cases}$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$ (Diffusion Kernel)

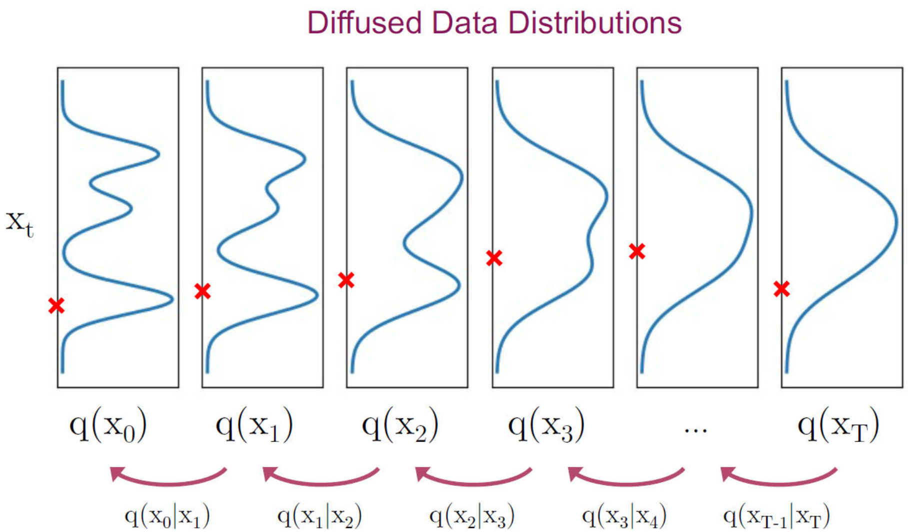
For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\ \mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\ \epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

β_t values schedule (i.e., the noise schedule) is designed such that $\bar{\alpha}_T \to 0$ and $q(\mathbf{x}_T|\mathbf{x}_0) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Denoising Diffusion Models:

Learning to generate by denoising (cont'd)

- Generative learning by denoising
  - Diffusion parameters are designed such that: $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I}))$



Generation:

Sample $\mathbf{x}_T \sim \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

Iteratively sample $\mathbf{x}_{t-1} \sim \underbrace{q(\mathbf{x}_{t-1}|\mathbf{x}_t)}_{\text{True Denoising Dist.}}$

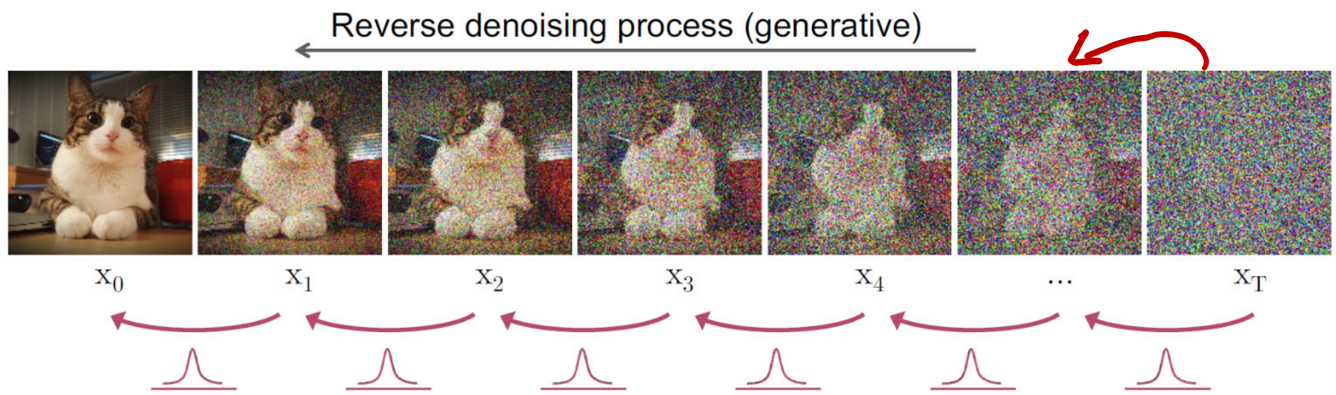
  - Unfortunately, $q(\mathbf{x}_{t-1}|\mathbf{x}_t) \propto q(\mathbf{x}_{t-1})q(\mathbf{x}_t|\mathbf{x}_{t-1})$ is intractable.
    We approximate $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ by Normal distribution by setting small β_t in each step

Denoising Diffusion Models:

Learning to generate by denoising (cont'd)

- Reverse diffusion process
  - Learn to denoise in T steps
  - **Let the model θ predict** $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$
  - And, to conclude the training process first, we need to predict the **noise** added in image.

Data



Noise

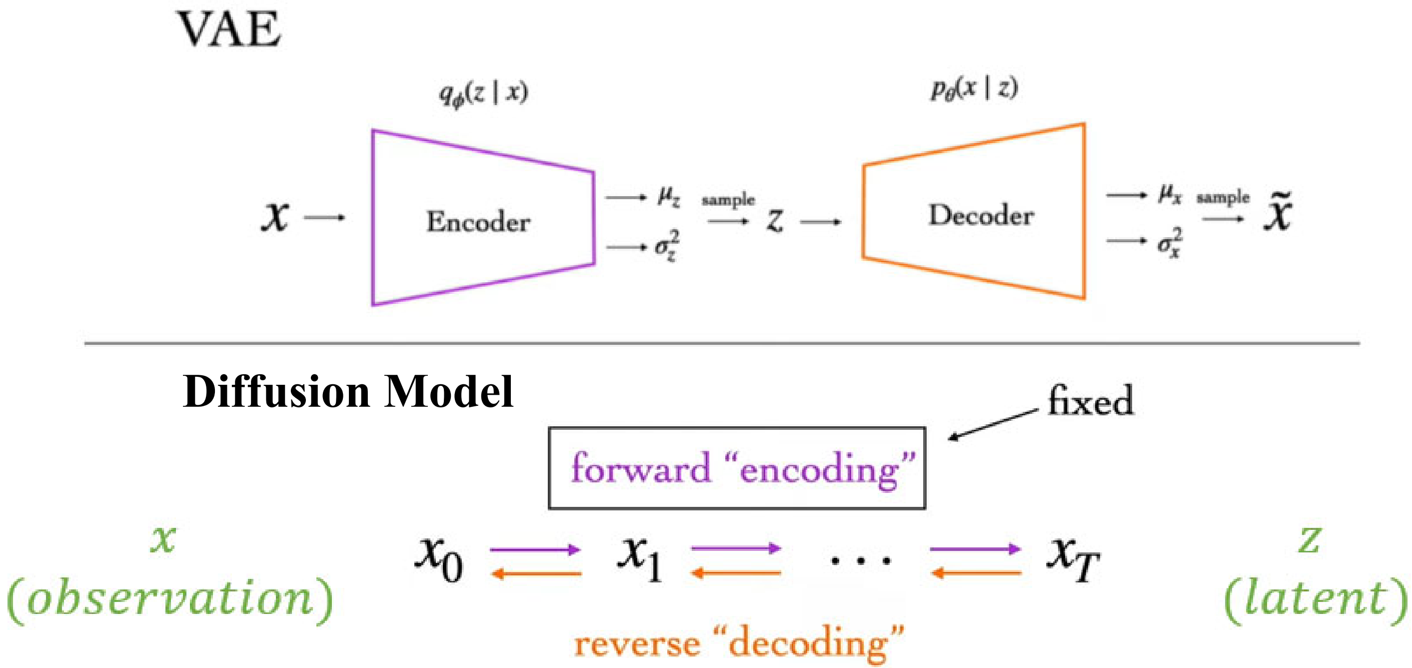
$$p(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \underbrace{\boxed{\mu_\theta(\mathbf{x}_t, t)}}_{\substack{\text{Trainable network} \\ \text{(U-net, Denoising Autoencoder)}}}, \sigma_t^2\mathbf{I})$$

➡ $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$

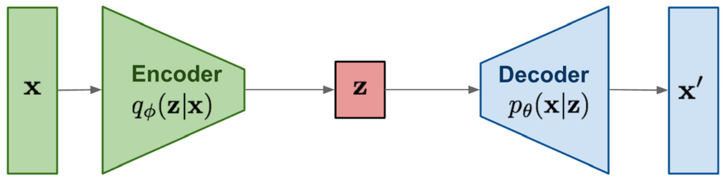
Learning of Diffusion Models

- $\boxed{\log p_\theta(x) \geq \text{ variational lower bound}}$



$$\log p_\theta(x) \geq \text{ variational lower bound}$$

Recall: Training VAE



Enc

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z)p(z)}{p_\theta(z \mid x)} = \log \frac{p_\theta(x|z)p(z)q_\phi(z|x)}{p_\theta(z|x)q_\phi(z|x)}$$

1 2 3

$$= E_z[\log p_\theta(x|z)] - E_z\left[\log \frac{q_\phi(z|x)}{p(z)}\right] + E_z\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$$

N

$$= E_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x), p(z)\right) + D_{KL}(q_\phi(z|x), p_\theta(z|x))$$

Data reconstruction

KL divergence between sample distribution from the encoder and the prior

Enc.

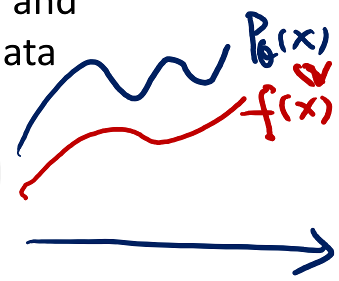
KL divergence between sample distribution from the encoder and the posterior of data

≥ 0

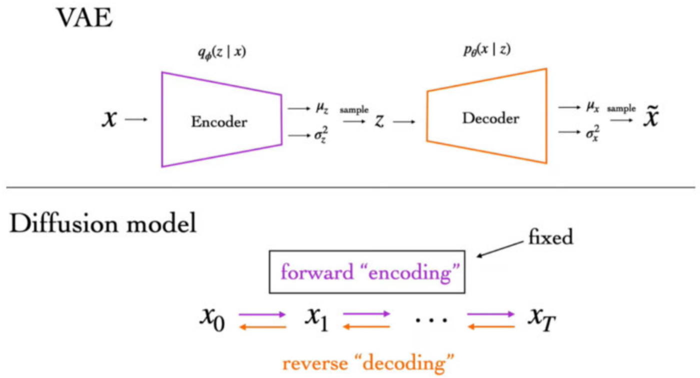
f(x)

$$\log p_\theta(x) \geq E_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x), p(z)\right)$$

1 2



i.e., **variational lower bound** on the **data likelihood $p_\Theta(x)$**



Learning of Diffusion Models

- $\boxed{\log p_\theta(x) \geq \text{ variational lower bound}}$
    - Recall that we exploit variational bound for optimizing VAE models

$$\log p_\theta(x) \geq E_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x), p(z)\right)$$

vs. $$\mathbb{E}_{q(\mathbf{x}_0)}\left[-\log p_\theta(\mathbf{x}_0)\right] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[-\log \frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] =: L$$

- In Ho et al. NeurIPS'20, it is shown that

$$\mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \beta I)$$

$$L = \mathbb{E}_q\left[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)||p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)||p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{-\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\right]$$

$$\mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I)$$

$$\tilde{\mu}_t(x_t, x_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon\right)$$

fixed

from slide #9

Ref: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/
https://youtu.be/fbLgFrlTnGU
https://youtu.be/HoKDTa5jHvg

Learning of Diffusion Models (cont'd)

- Recall that

Not trainable $\qquad\qquad\qquad\qquad$ {0, 1, …, 255] → [-1,1]

$$L = \mathbb{E}_q\Big[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0)\,\|\,p(\mathbf{x}_T))}_{L_T} + \sum_{t>1}\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)\,\|\,p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\Big]$$

- Still working on it…
  - Only care about KL divergence between two Gaussian distributions

$$\begin{cases} q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1},\mathbf{x}_0)\dfrac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} := \mathcal{N}(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I) \\ p_\theta(x_{t-1}\,|\,x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \end{cases}$$

learned $\qquad$ fixed

$$\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon\right) \quad (1)$$

(actual μ)

$$\frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\varepsilon_\theta(x_t, t)\right) \quad (2)$$

(predicted μ)

  - As a result,

$$\frac{1}{2\sigma_t^2}\|(1)-(2)\|^2 \longrightarrow \mathbb{E}_{\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2\alpha_t(1-\bar{\alpha}_t)}\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

  - For simplicity, we calculate

$$L_{\mathrm{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\right\|^2\right]$$

Slide credit: Kreis, Gao, & Vahdat
and https://youtu.be/HoKDTa5jHvg

Learning of Diffusion Models

- Summary
  - Training and sample generation

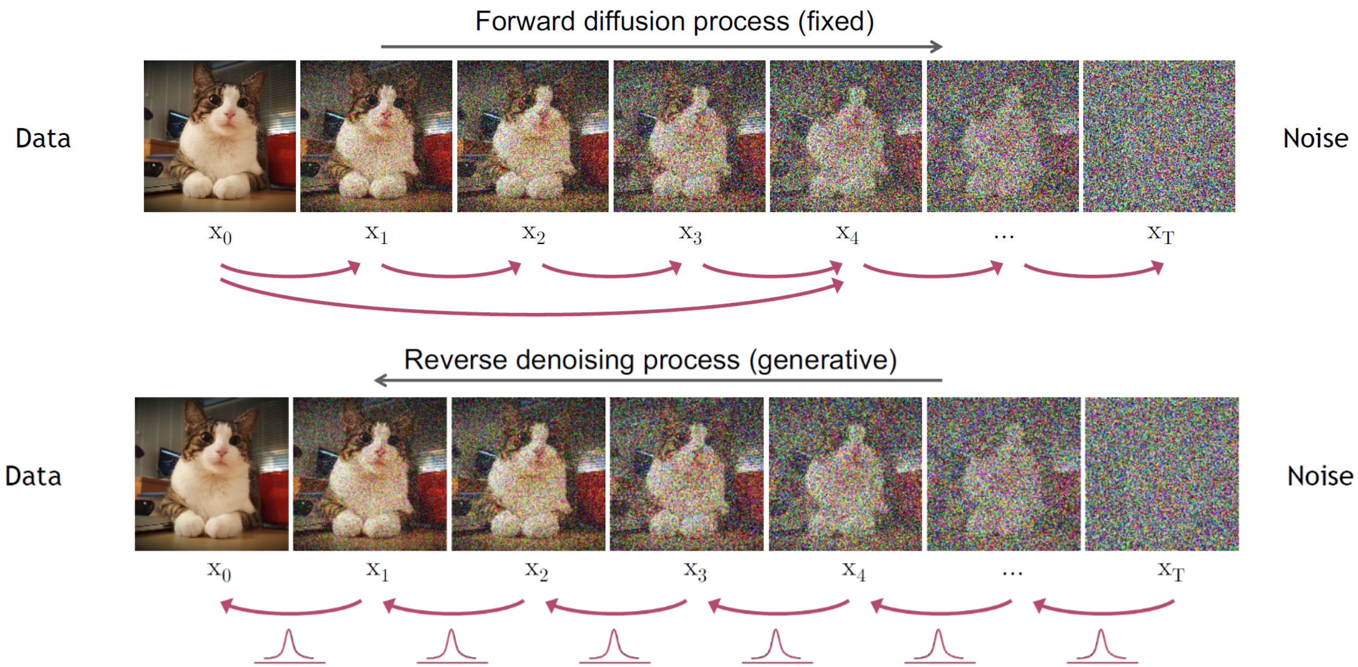
Inference.

Algorithm 1 Training

repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
  $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  Take gradient descent step on
  $$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}, t) \right\|^2$$
until converged

Algorithm 2 Sampling

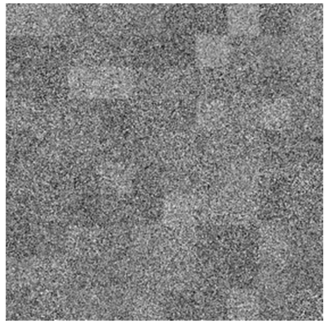
$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
for $t = T, \ldots, 1$ **do**
  $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  $$\mathbf{x}_{t-1} = \boxed{\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)} + \sigma_t \mathbf{z}$$
end for
return $\mathbf{x}_0$



Slide credit: Kreis, Gao, & Vahdat

Learning of Diffusion Models

- Summary
  - Training and sample generation



E.g., MNIST

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta\left(\boxed{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}}, t\right) \right\|^2$$
6: **until** converged

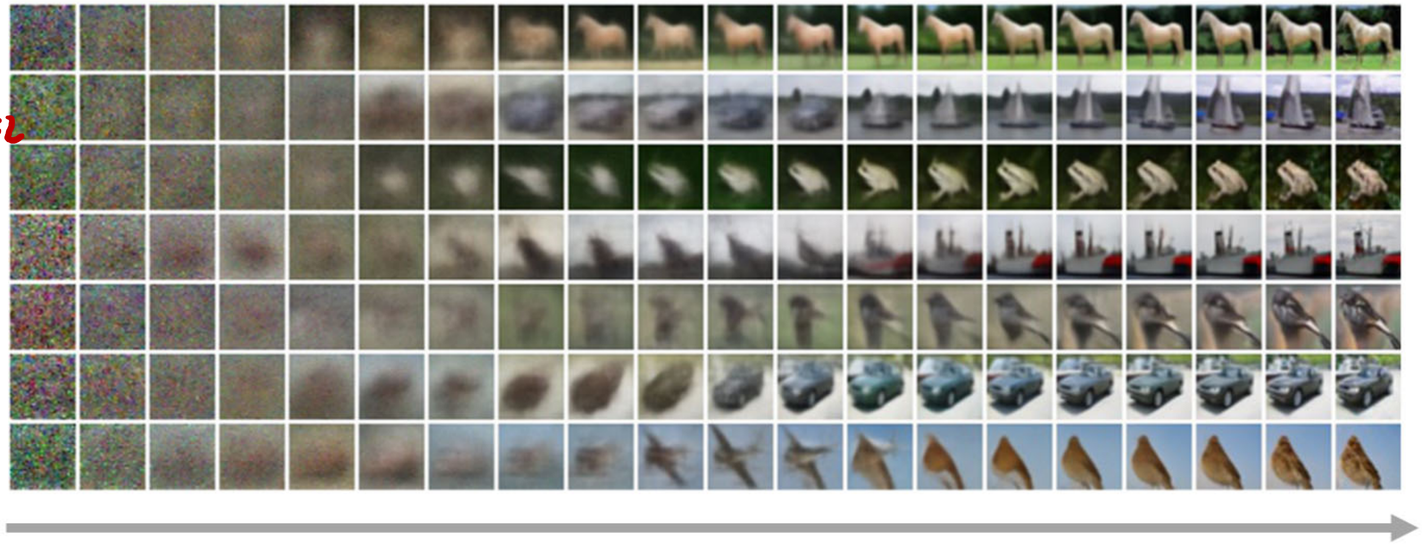
Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
4: $\mathbf{x}_{t-1} = \boxed{\frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right)} + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

z_1

x_T z_2

x_0

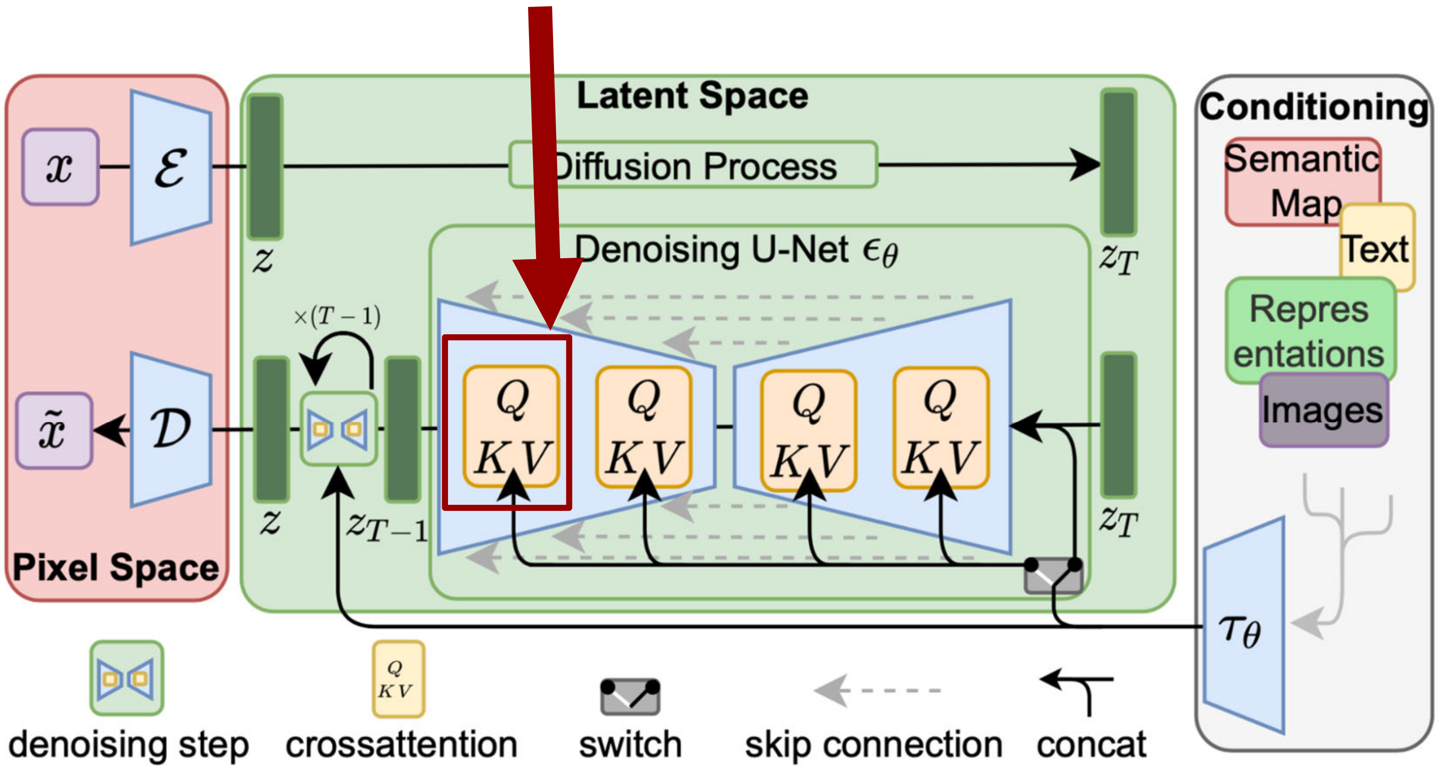


More steps

Slide credit: Kreis, Gao, & Vahdat
https://medium.com/ai-blog-tw/%E9%82%8A%E5%AF%A6%E4%BD%9C%E9%82%8A%E5%AD%B8%E7%BF%92diffusion-model-%E5%BE%9Eddpm%E7%9A%84%E7%B0%A1%E5%8C%96%E6%A6%82%E5%BF%B5%E7%90%86%E8%A7%A3-4c565a1c09c

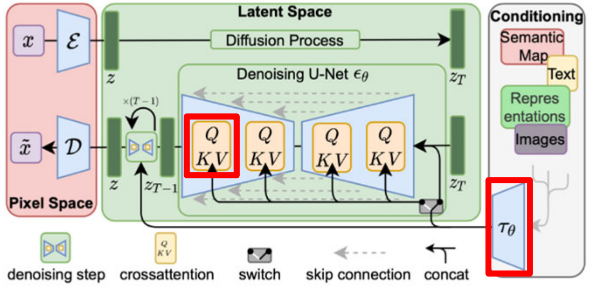
From Unconditional to Conditional Latent Diffusion Model

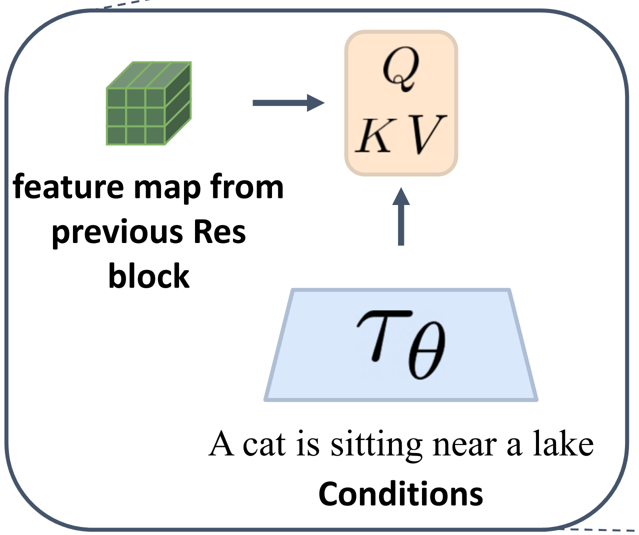
- Condition mechanism: using transformer layers

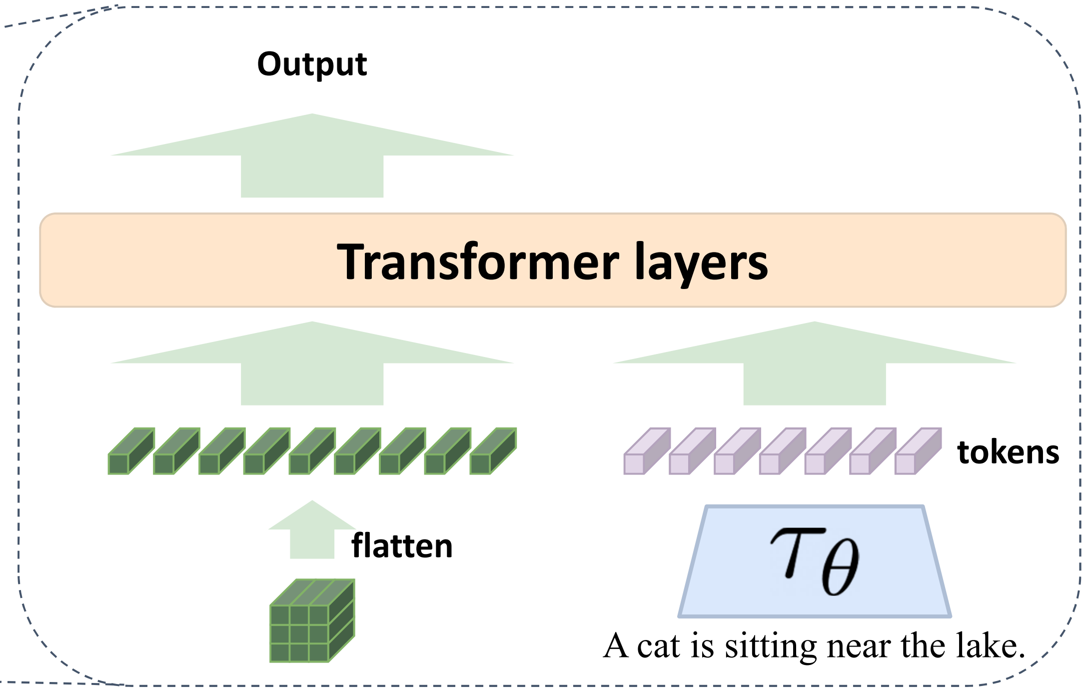


From Unconditional to Conditional Latent Diffusion Model (cont'd)

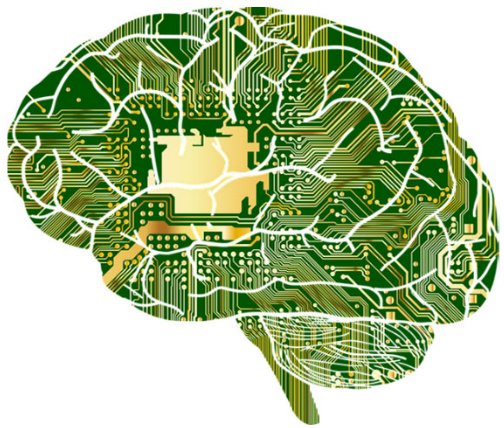
- **Condition mechanism:** using transformer layers
- τ_θ is embedding module for conditions, e.g. BERT.





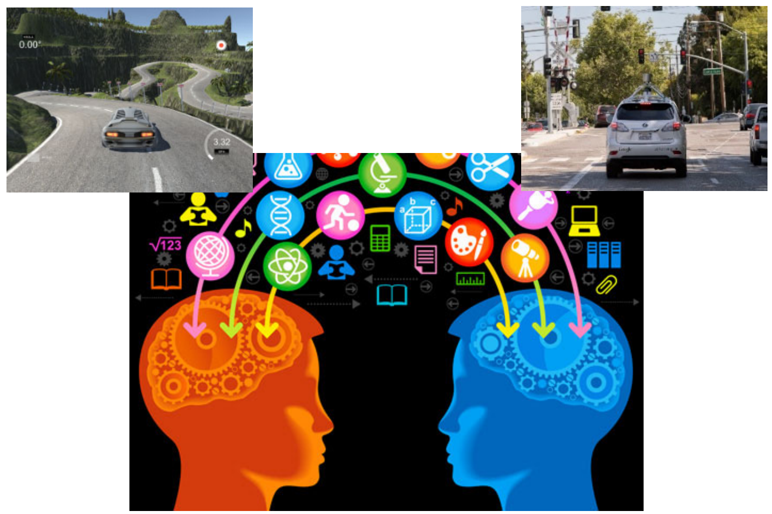


What to Be Covered Today…

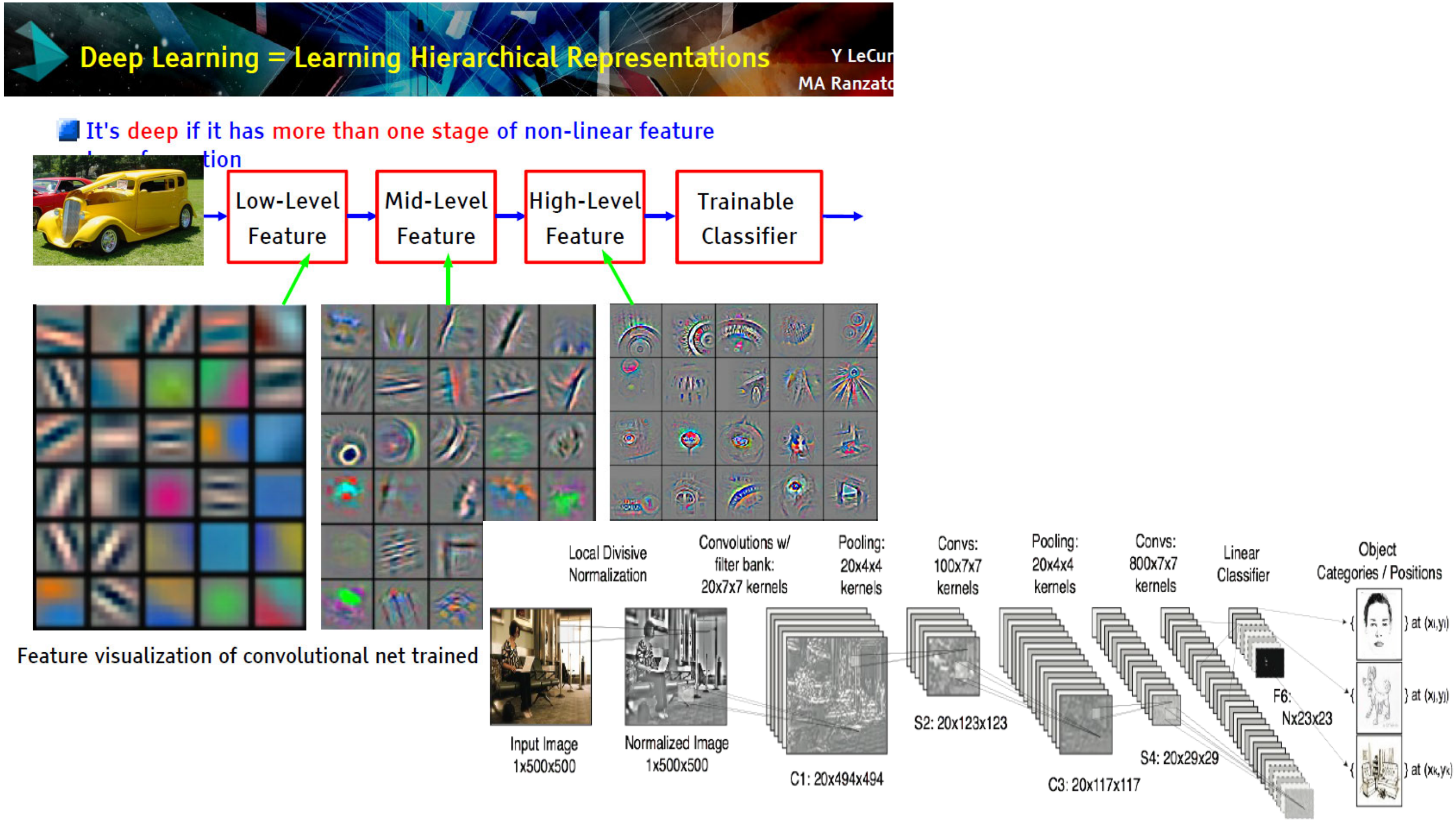


- Generative Models
  - Diffusion Model
- Transfer Learning
  - Visual Classification – Domain Adaptation
  - Visual Synthesis – Style Transfer
- Representation Disentanglement
  - Supervised vs. unsupervised feature disentanglement

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.



Revisit of CNN for Visual Classification



LeCun & Ranzato, Deep Learning Tutorial, ICML 2013

(Traditional) Machine Learning vs. Transfer Learning

- Machine Learning
  - Collecting/annotating data is typically expensive.

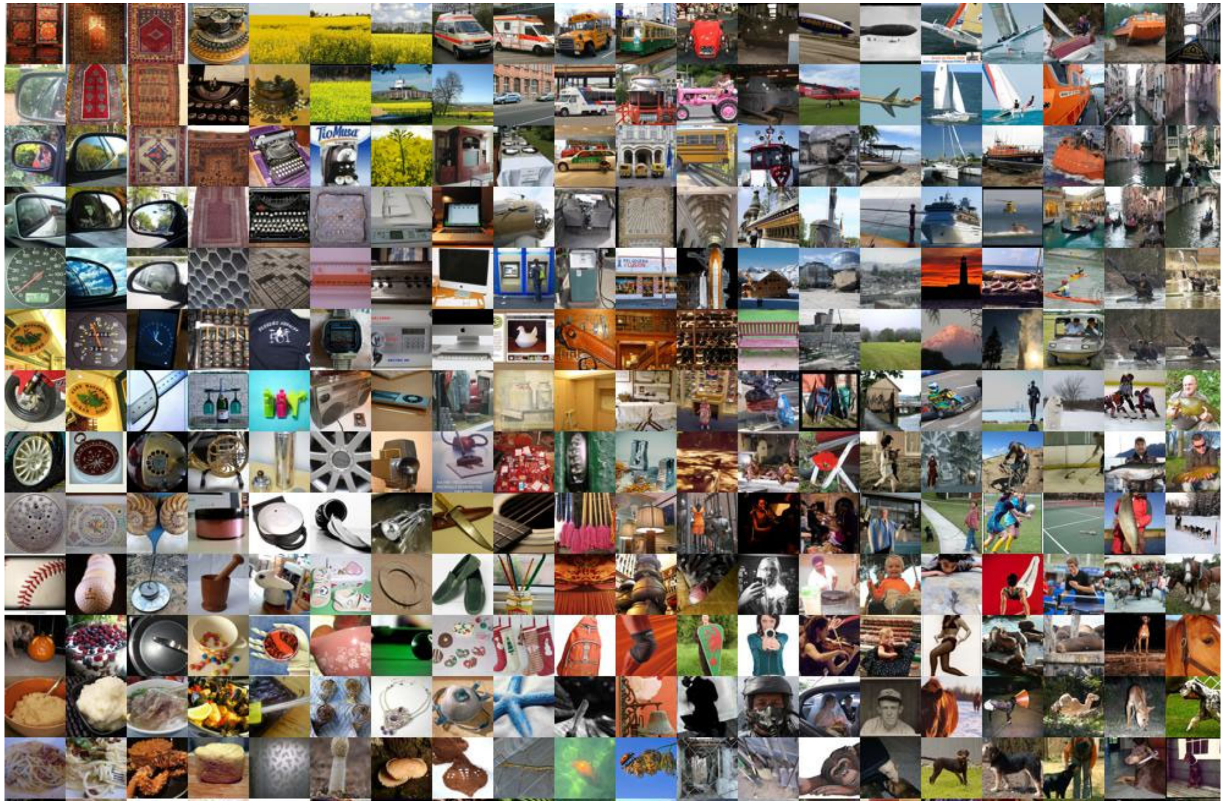
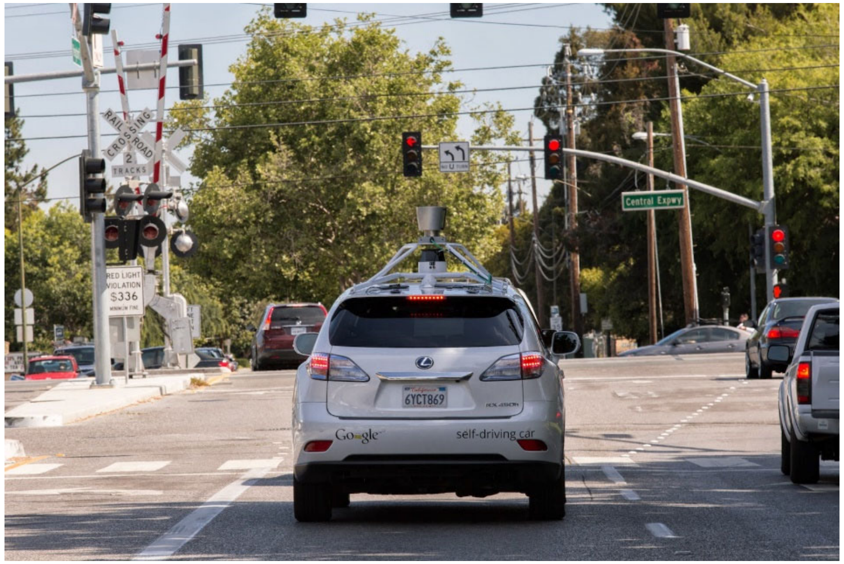


Image Credit: A. Karpathy

Transfer Learning: What, When, and Why? (cont'd)

- A More Practical Example



training: source domain $\{X_s, Y_s\}$ $\longleftrightarrow$ target domain $\{X_T\}$ $\rightarrow$ UDA

domain diff (domain bias)

https://techcrunch.com/2017/02/08/udacity-open-sources-its-self-driving-car-simulator-for-anyone-to-use/
https://googleblog.blogspot.tw/2014/04/the-latest-chapter-for-self-driving-car.html

Domain Adaptation in Transfer Learning

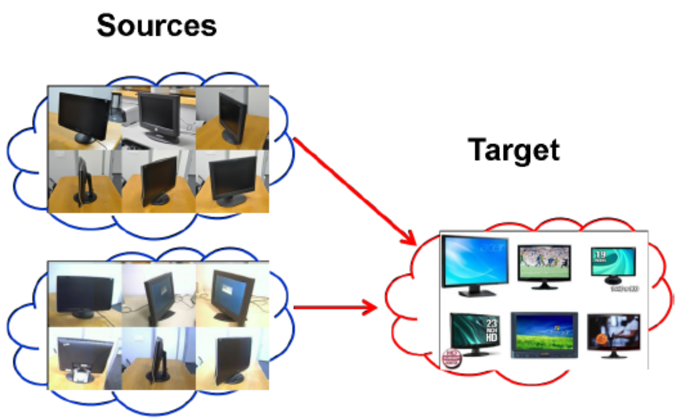
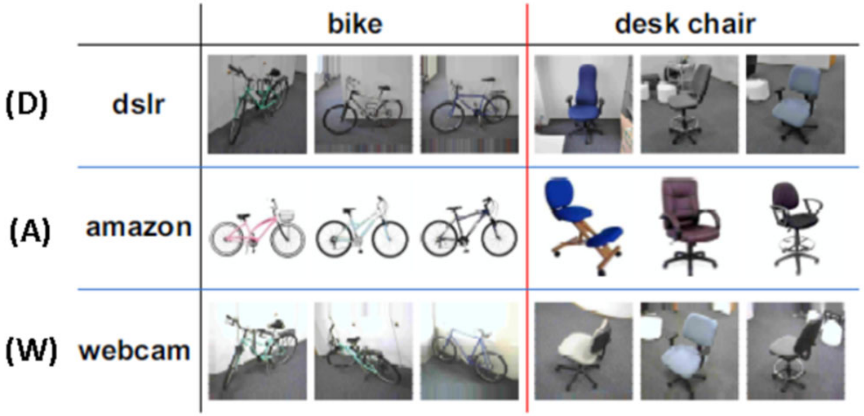


Image: Courtesy to S.J. Pan

- What's DA?
  - Leveraging info source to target domains, so that the same learning task across domains (or particularly in the target domain) can be addressed.
  - Typically all the source-domain data are labeled.
- Settings
  - Semi-supervised DA: few target-domain data are with labels.
  - Unsupervised DA: no label info available in the target-domain. (shall we address supervised DA?)
  - Imbalanced DA: fewer classes of interest in the target domain
  - Homogeneous vs. heterogeneous DA

Deep Feature is Sufficiently Promising.

- DeCAF
  - Leveraging an auxiliary large dataset to train CNN.
  - The resulting features exhibit sufficient representation ability.
  - Supporting results on Office+Caltech datasets, etc.



Feature	SURF										$Decaf_6$				
data	Raw	SA	SDA	GFK	TCA	JDA	TJM	SCA	JGSA primal	JGSA linear	JGSA RBF	JDA	OTGL	JGSA primal	JGSA linear	JGSA RBF
A→D	35.67	33.76	33.76	40.13	33.76	39.49	45.22	39.49	**47.13**	45.86	45.22	81.53	85.00	**88.54**	85.35	84.71
A→W	31.19	33.22	30.85	36.95	36.27	37.97	42.03	34.92	45.76	**49.49**	45.08	80.68	83.05	81.02	**84.75**	80.00
D→A	28.29	**39.87**	38.73	28.71	31.00	33.09	32.78	31.63	38.00	36.01	38.73	91.96	**92.31**	91.96	92.28	91.96
D→W	83.73	76.95	76.95	80.34	86.10	89.49	85.42	84.41	91.86	91.86	**93.22**	99.32	96.29	**99.66**	98.64	98.64
W→A	31.63	39.25	39.25	27.56	28.91	32.78	29.96	29.96	39.87	**41.02**	40.81	90.71	90.62	90.71	**91.44**	91.34
W→D	84.71	75.16	75.80	85.35	89.17	89.17	89.17	87.26	**90.45**	**90.45**	88.54	**100**	96.25	**100**	**100**	**100**

Donahue et al., DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition, ICML 2014

Recent Deep Learning Methods for DA

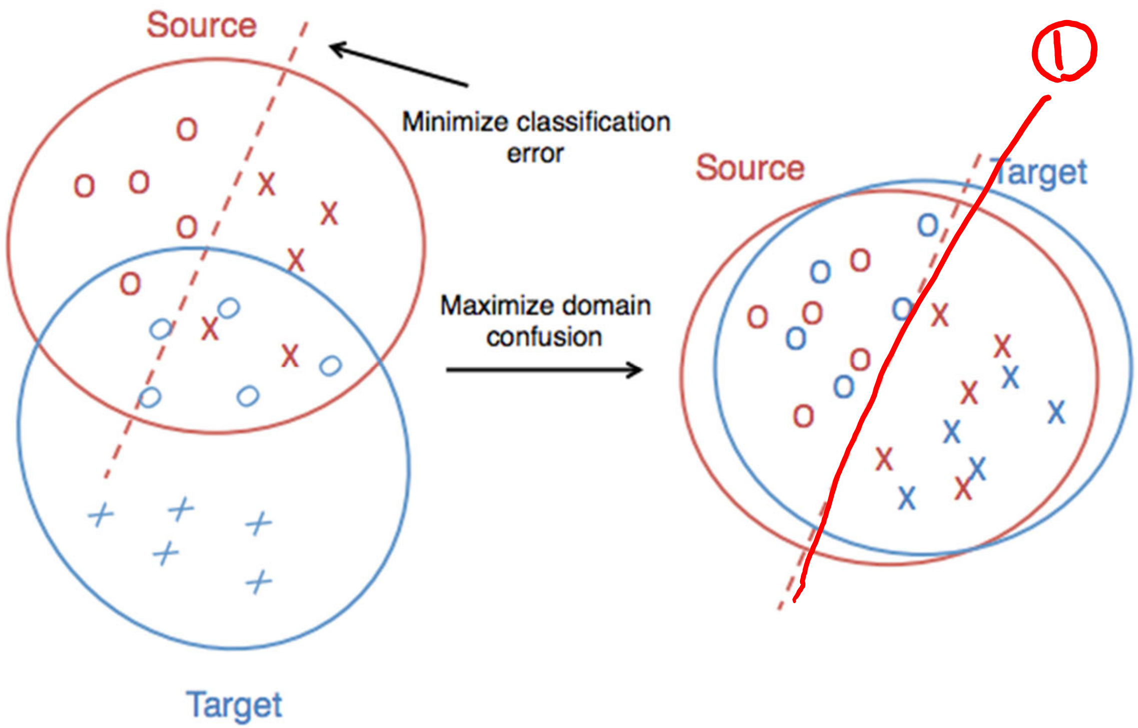
- Deep Domain Confusion (DDC)
- Domain-Adversarial Training of Neural Networks (DANN)
- Adversarial Discriminative Domain Adaptation (ADDA)
- Domain Separation Network (DSN)
- Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks (PixelDA)
- No More Discrimination: Cross City Adaptation of Road Scene Segmenters

	Shared weights	Adaptation loss	Generative model
DDC	✓	MMD	✘
DANN	✓	Adversarial	✘
ADDA	✘	Adversarial	✘
DSN	Partially shared	MMD/Adversarial	✘
PixelDA	✘	Adversarial	✓

Deep Domain Confusion (DDC)

Source $\{X_S, Y_S\}$
tgt $\{X_T\}$
CNN

- Deep Domain Confusion: Maximizing for Domain Invariance
  - Tzeng et al., arXiv: 1412.3474, 2014



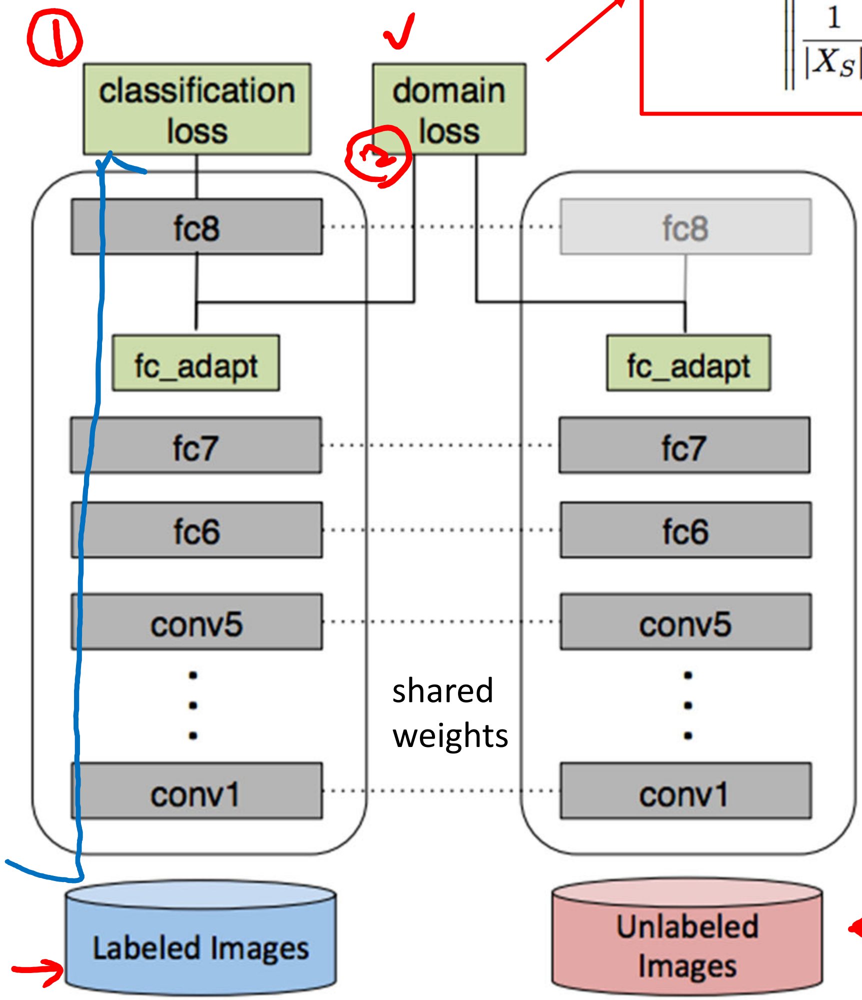
Deep Domain Confusion (DDC)

fc7

$$\mathrm{MMD}(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \phi(x_t) \right\|$$

train: ① + ②

testing?



{Xs, Ys}

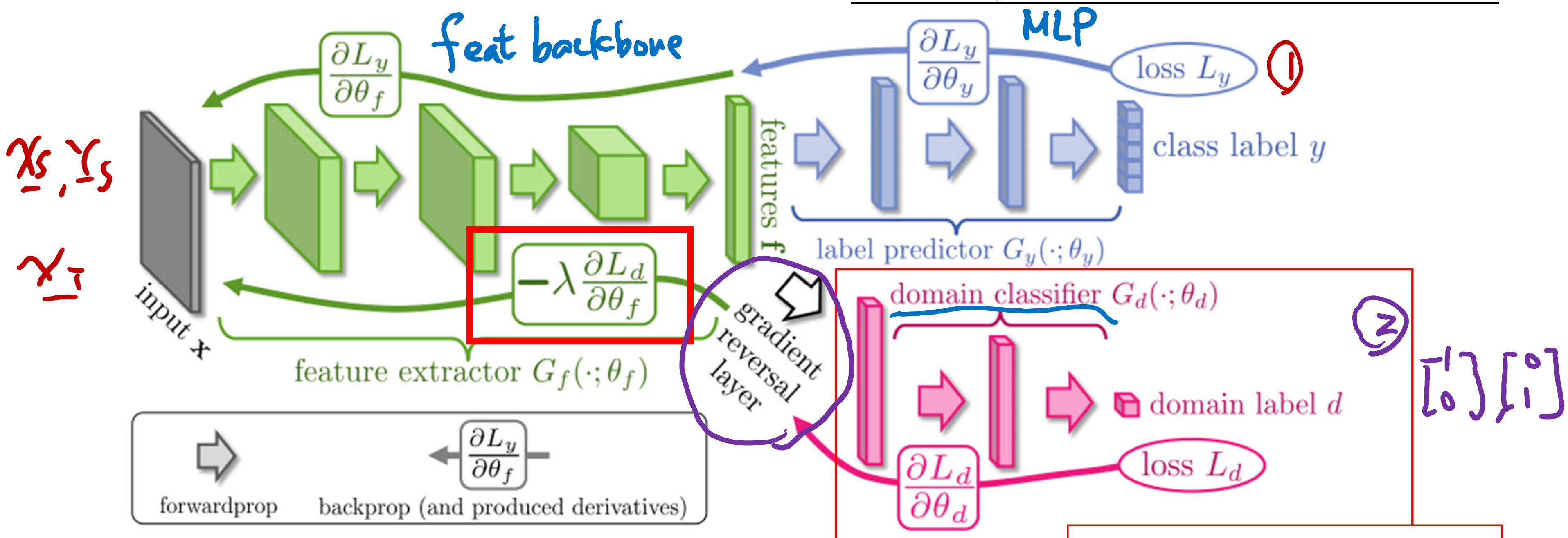
{XT}

✓Minimize classification loss:

$$\mathcal{L} = \mathcal{L}_C(X_L, y) + \lambda\,\mathrm{MMD}^2(X_S, X_T)$$

Domain Confusion by Domain-Adversarial Training

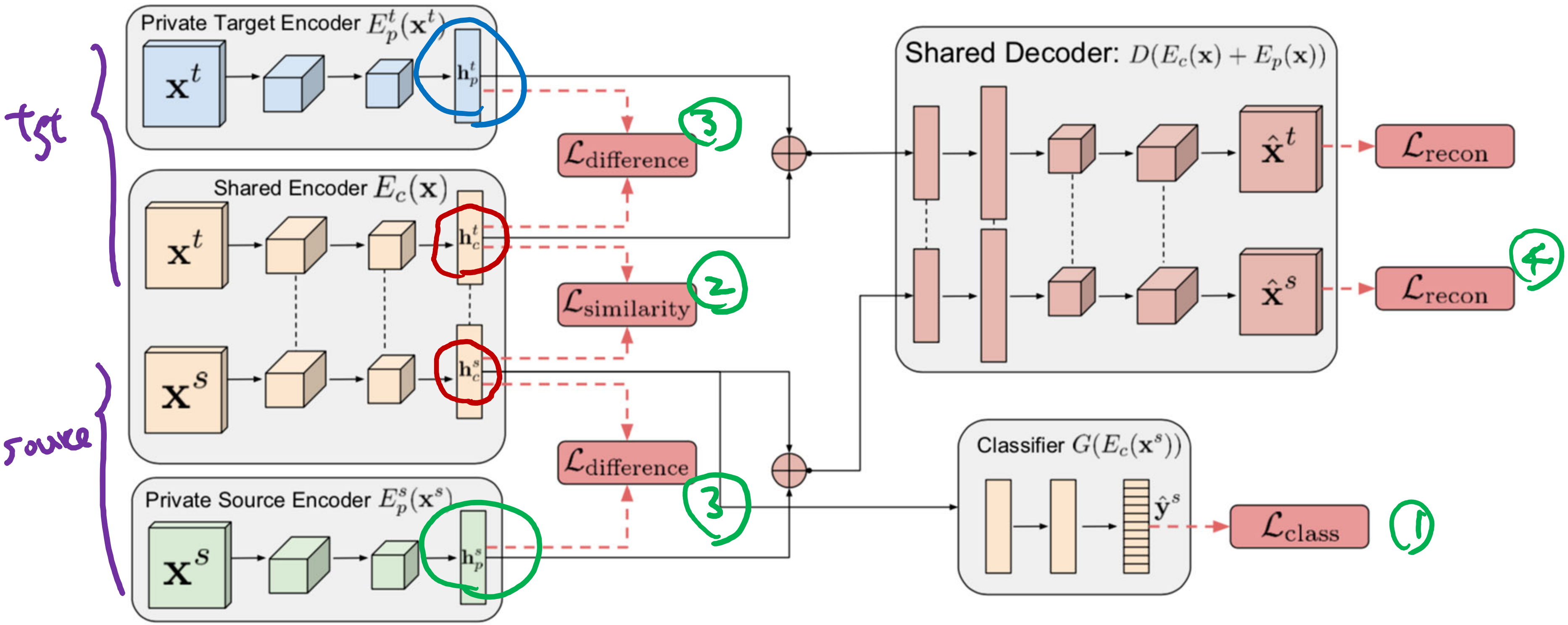
- Domain-Adversarial Training of Neural Networks (DANN)
  - Y. Ganin et al., ICML 2015
  - Maximize domain confusion = maximize domain classification loss
  - Minimize source-domain data classification loss
  - The derived feature f can be viewed as a disentangled & domain-invariant feature.



$$d_i \log \hat{d}_i \; + \; (1 - d_i) \log(1 - \hat{d}_i)$$

Beyond Domain Confusion

- **Domain Separation Network (DSN)**
  - Bousmalis et al., NIPS 2016
  - Separate encoders for domain-invariant and domain-specific features
  - Private/common features are *disentangled* from each other.



Beyond Domain Confusion

- Domain Separation Network, NIPS 2016
  - Example results

Source-domain image X_s

Reconstruct private + shared features $D(E_c(x_s)+E_p(x_s))$

Reconstruct private feature $D(E_p(x_s))$

(a) MNIST (source) (b) MNIST-M (target)

Reconstruct shared feature only $D(E_c(x_s))$

Target-domain image X_T

Beyond Domain Confusion

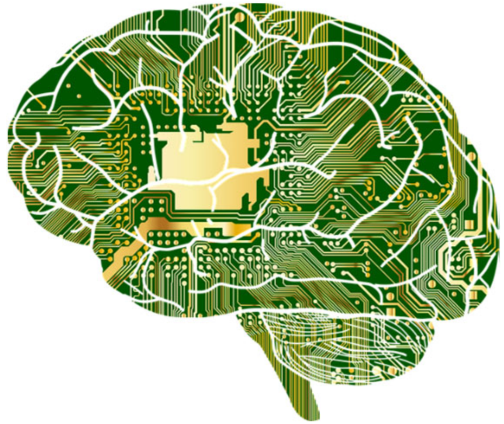
- Domain Separation Network, NIPS 2016
  - Example results

Source-domain image X_s

(a) MNIST (source) (b) MNIST-M (target)

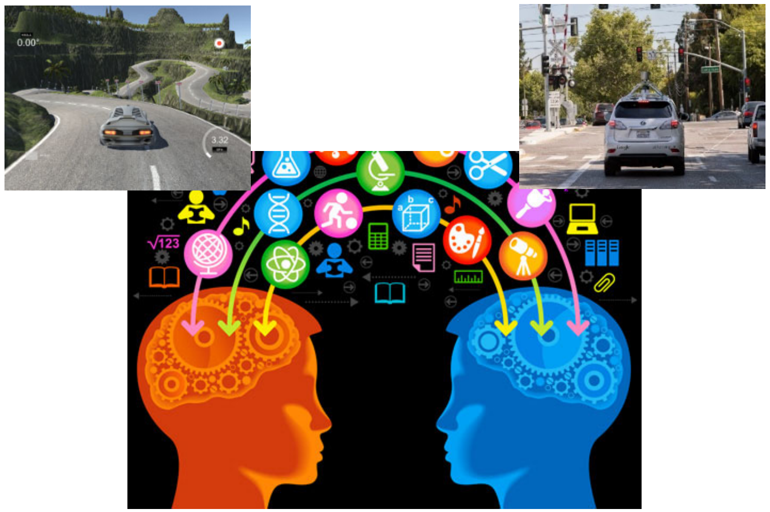
Target-domain image X_T

What to Be Covered Today…



- Generative Models
  - Diffusion Model
- Transfer Learning
  - Visual Classification – Domain Adaptation
  - Visual Synthesis – Style Transfer
- Representation Disentanglement
  - Supervised vs. unsupervised feature disentanglement

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.

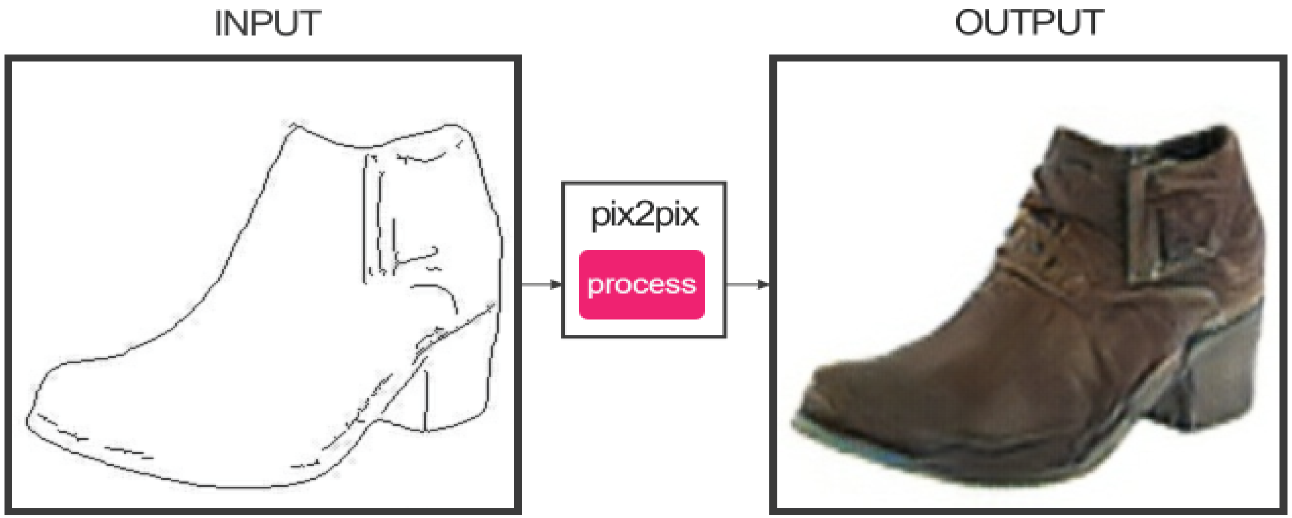


Transfer Learning for Image Synthesis

- Cross-Domain Image Translation
  - Pix2pix: Pairwise cross-domain training data
  - CycleGAN/DualGAN/DiscoGAN: Unpaired cross-domain training data
  - UNIT: Learning cross-domain image representation (with unpaired training data)
  - AdaIN: Single-image arbitrary style transfer in real-time
  - Beyond image translation

Pix2pix

- Image-to-image translation with conditional adversarial networks (CVPR’17)
  - Can be viewed as image style transfer



Sketch Photo

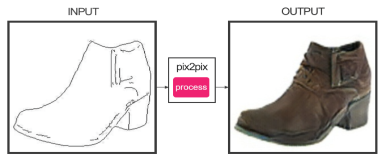
Isola et al. " Image-to-image translation with conditional adversarial networks." *CVPR* 2017.

Pix2pix

$\{\tilde{x}, x_{in}\} \rightarrow \boxed{D} \rightarrow T/F$

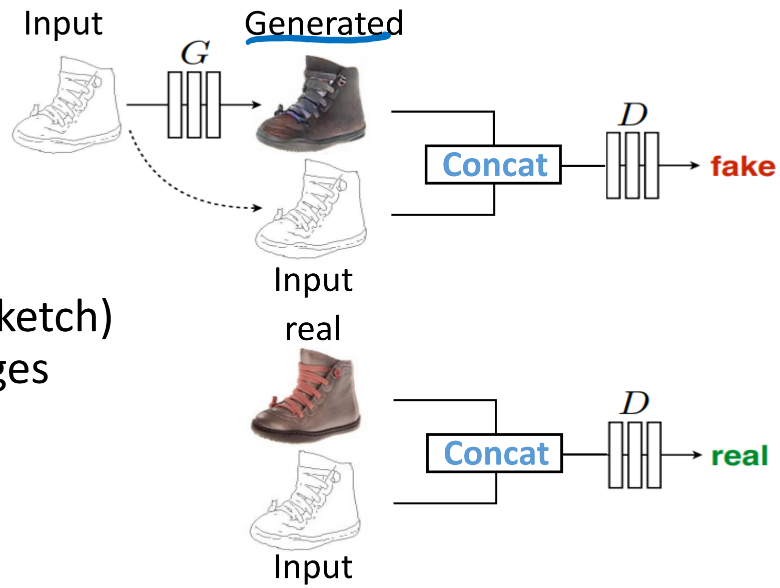
$\{x_{real}, x_{in}\} \rightarrow$

Testing Phase



- **Goal / Problem Setting**
  - Image translation across two distinct domains (e.g., sketch v.s. photo)
  - ✓ Pairwise training data
- Method: Conditional GAN
  - Example: Sketch to Photo
    - Generator
      Input: Sketch
      Output: Photo
    - Discriminator
      Input: Concatenation of Input(Sketch) & Synthesized/Real(Photo) images
      Output: Real or Fake

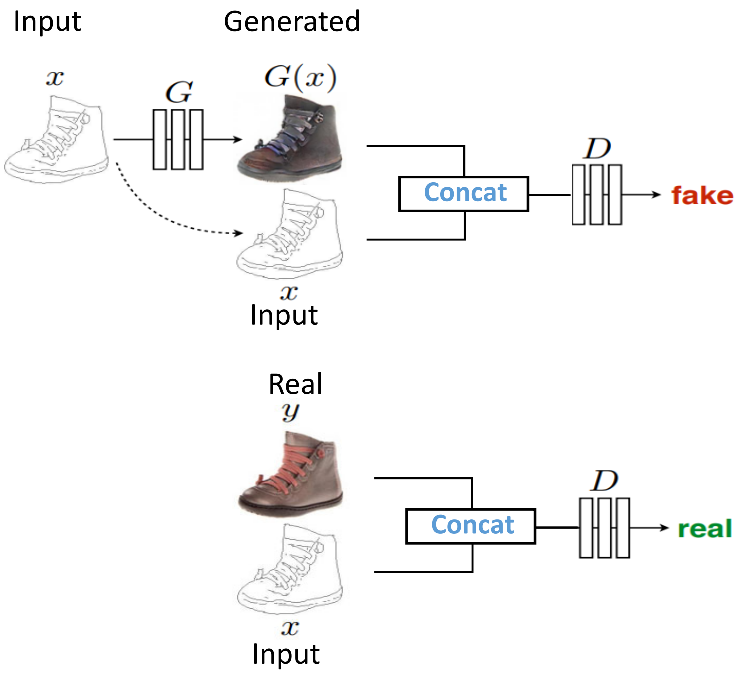
Training Phase



Pix2pix

- ***Learning the model***

Training Phase



Overall objective function

$$G^* = \arg\min_{G}\max_{D} \mathcal{L}_{cGAN}(G, D) + \mathcal{L}_{L1}(G)$$

Conditional GAN loss

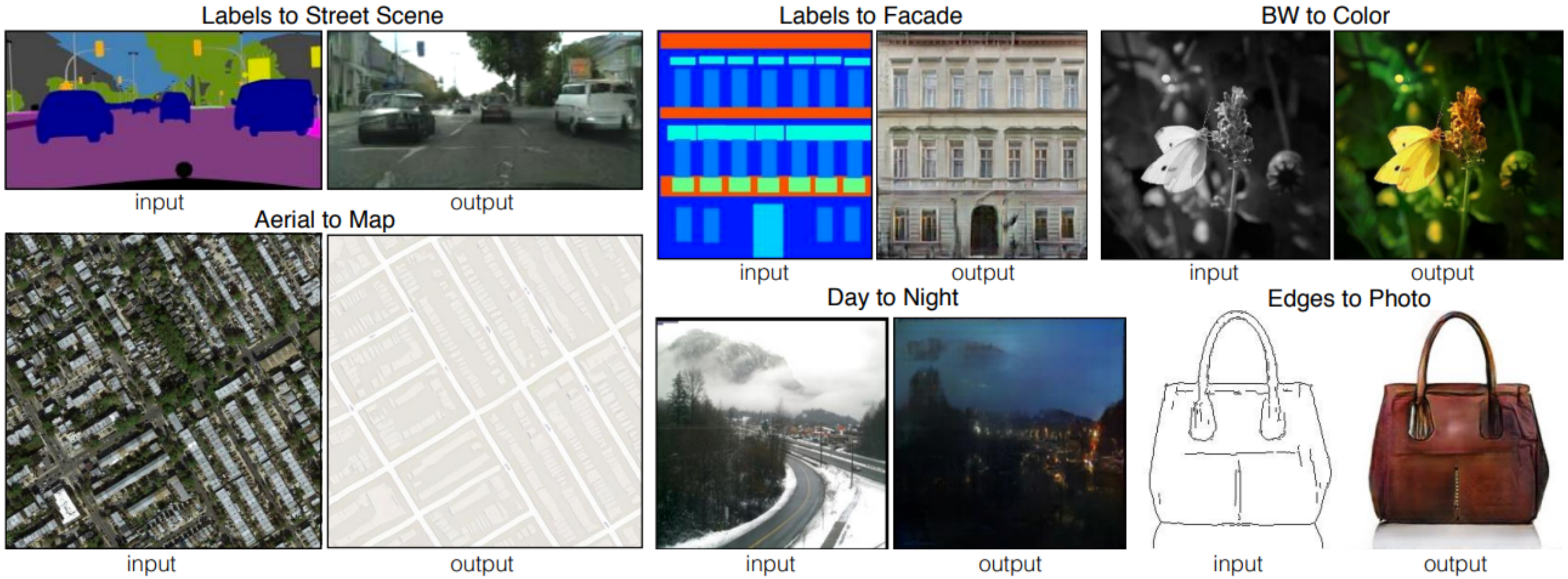
$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x}[\underbrace{\log(1 - D(\overbrace{x, G(x)}^{\text{Concatenate}}))]}_{\text{Fake (Generated)}} + \mathbb{E}_{x,y}[\underbrace{\log D(\overbrace{x, y}^{\text{Concatenate}})]}_{\text{Real}}$$

Reconstruction Loss

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y}[\|y - G(x)\|_1]$$

Pix2pix

- ***Experiment results***



Demo page: https://affinelayer.com/pixsrv/

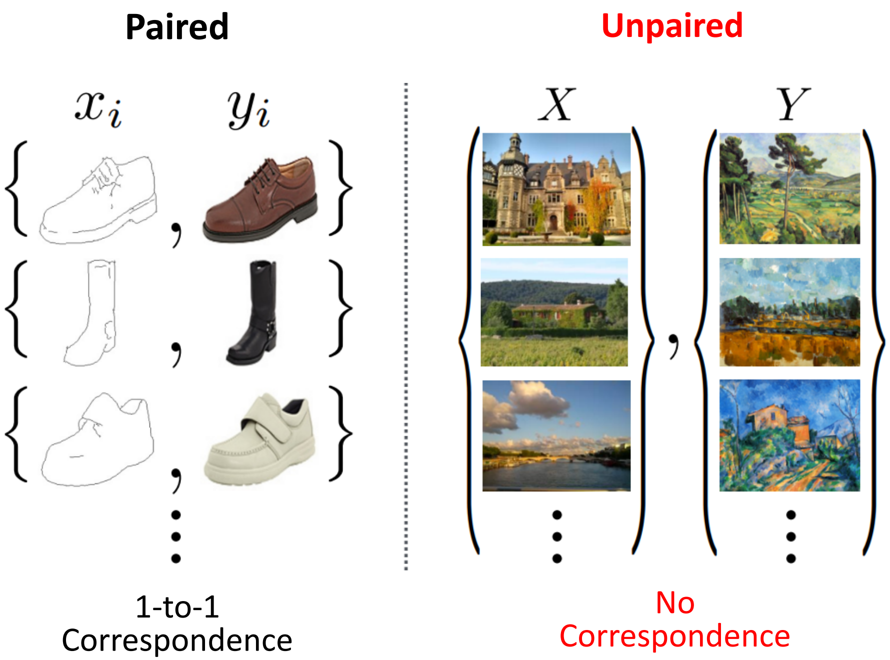
Isola et al. " Image-to-image translation with conditional adversarial networks." *CVPR* 2017.

Transfer Learning for Image Synthesis

- Cross-Domain Image Translation
  - Pix2pix: Pairwise cross-domain training data
  - CycleGAN/DualGAN/DiscoGAN: Unpaired cross-domain training data
  - UNIT: Learning cross-domain image representation (with unpaired training data)
  - AdaIN
  - Beyond image translation

CycleGAN/DiscoGAN/DualGAN

- CycleGAN
  - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks -to-image translation with conditional adversarial networks



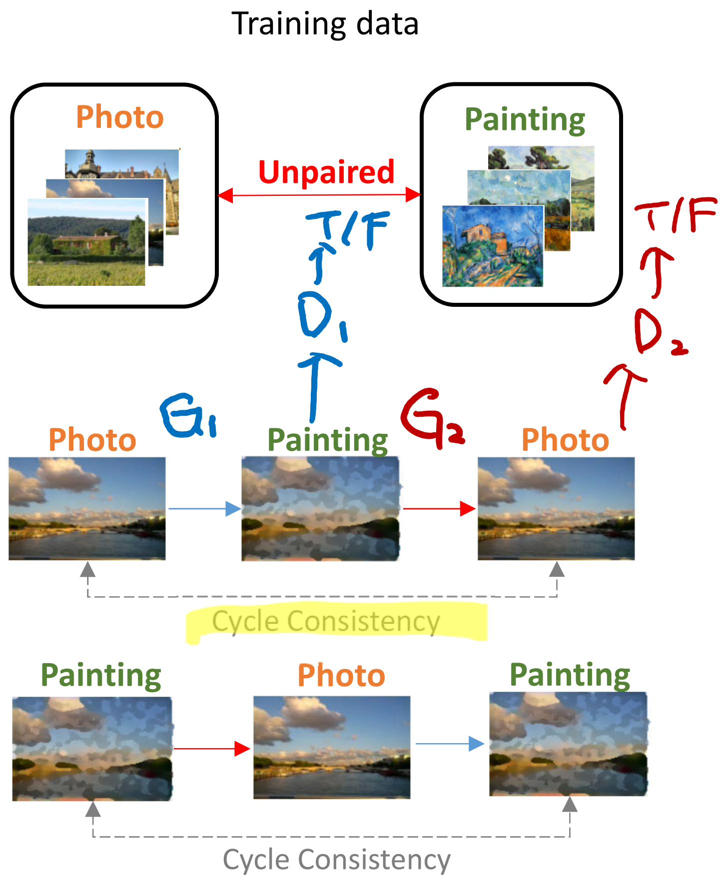
- Easier to collect training data
- More practical

Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

CycleGAN

- **Goal / Problem Setting**
  - Image translation across two distinct domains
  - Unpaired training data

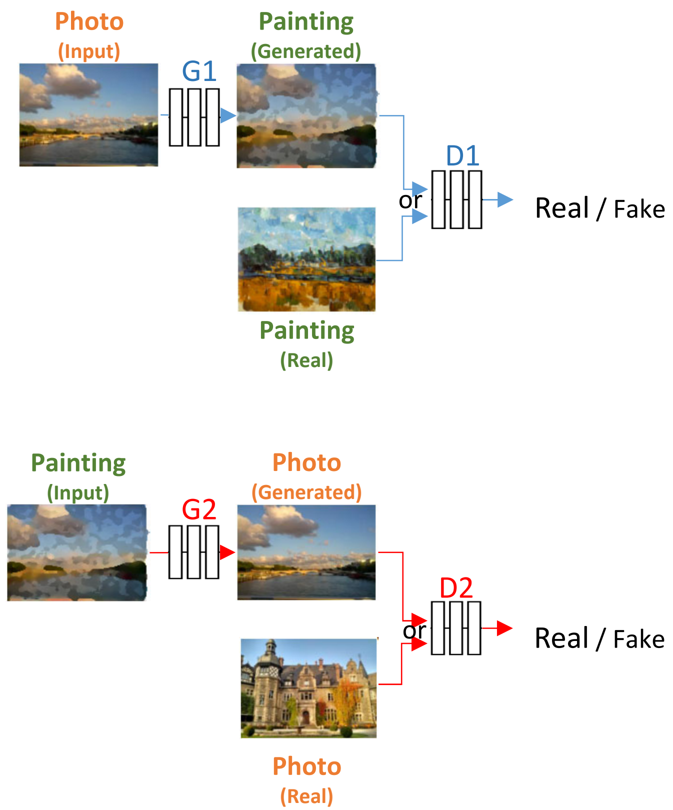
- **Idea**
  - Autoencoding-like image translation
  - Cycle consistency between two domains



Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

CycleGAN

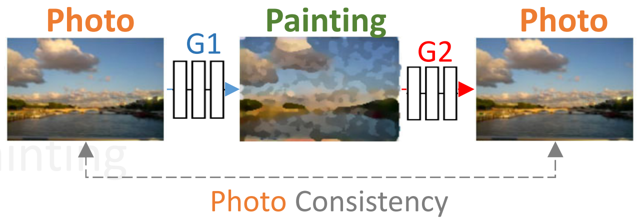
- Method (Example: Photo & Painting)
  - Based on 2 GANs
    - First GAN (G1, D1): Photo to Painting
    - Second GAN (G2, D2): Painting to Photo
  - Cycle Consistency
    - Photo consistency
    - Painting consistency



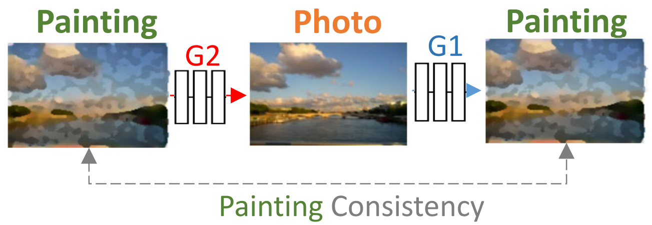
Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

CycleGAN

- Method (Example: Photo vs. Painting)
  - Based on 2 GANs
    - First GAN (G1, D1): Photo to Pa
    - Second GAN (G2, D2): Photo to Painting

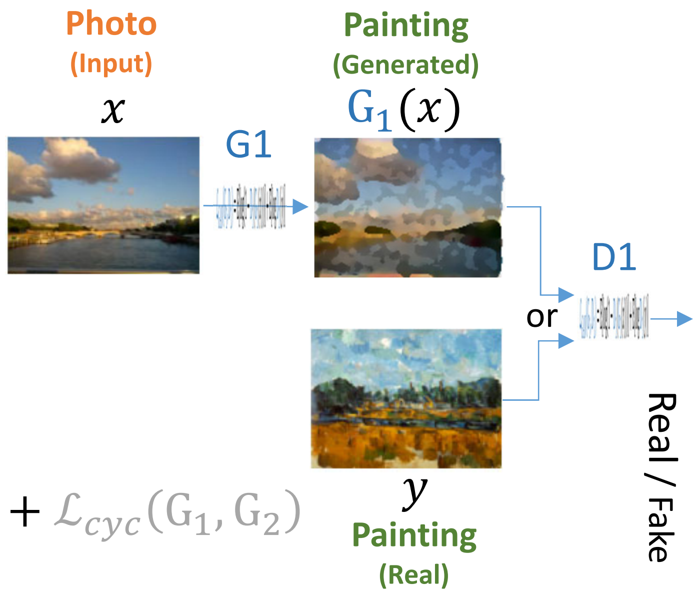


  - Cycle Consistency
    - Photo consistency
    - Painting consistency



Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

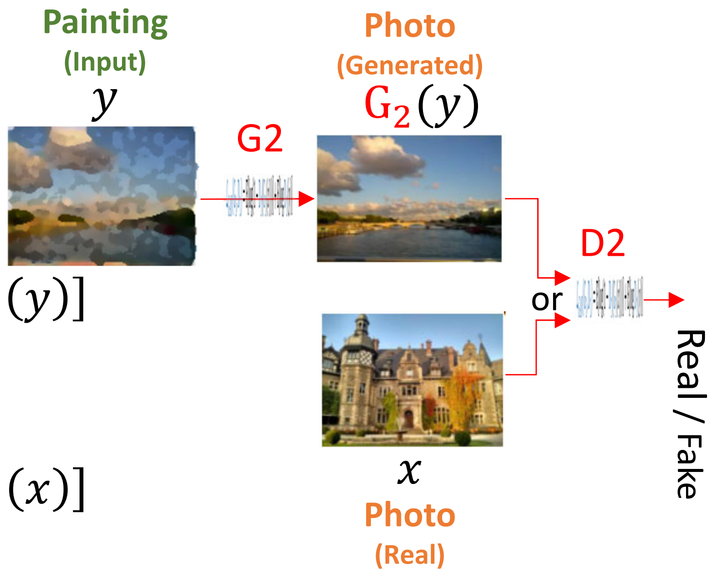
CycleGAN



- **Learning**

Overall objective function

$$G_1^*, G_2^* = \arg\min_{G_1, G_2}\max_{D_1, D_2} \underbrace{\mathcal{L}_{GAN}(G_1, D_1)}_{\text{First GAN}} + \underbrace{\mathcal{L}_{GAN}(G_2, D_2)}_{\text{Second GAN}} + \mathcal{L}_{cyc}(G_1, G_2)$$



- **Adversarial Loss**
  - First GAN (G1, D1):

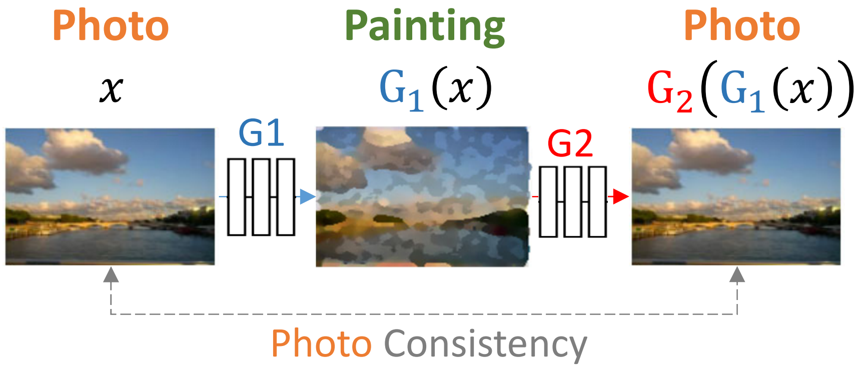
$$\mathcal{L}_{GAN}(G_1, D_1) = \mathbb{E}[\log(1 - D_1(G_1(x)))] + \mathbb{E}[\log D_1(y)]$$

  - Second GAN (G2, D2):

$$\mathcal{L}_{GAN}(G_2, D_2) = \mathbb{E}[\log(1 - D_2(G_2(y)))] + \mathbb{E}[\log D_2(x)]$$

Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

CycleGAN



- **Learning**

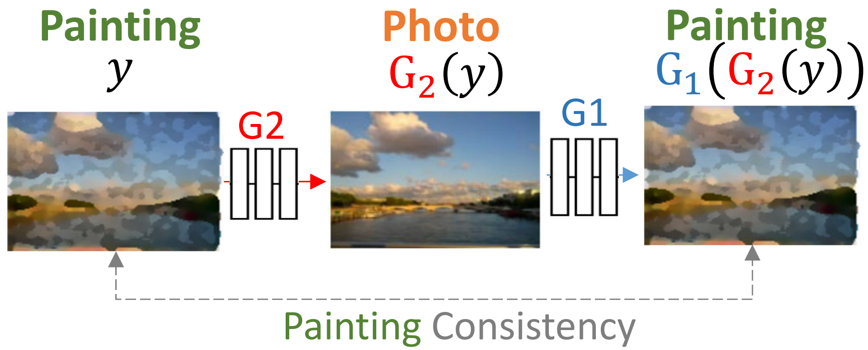
Overall objective function

$$G_1^*, G_2^* = \arg \min_{G_1, G_2} \max_{D_1, D_2} \mathcal{L}_{GAN}(G_1, D_1) + \mathcal{L}_{GAN}(G_2, D_2) + \mathcal{L}_{cyc}(G_1, G_2)$$

Cycle Consistency

- **Consistency Loss**
  - Photo and Painting consistency

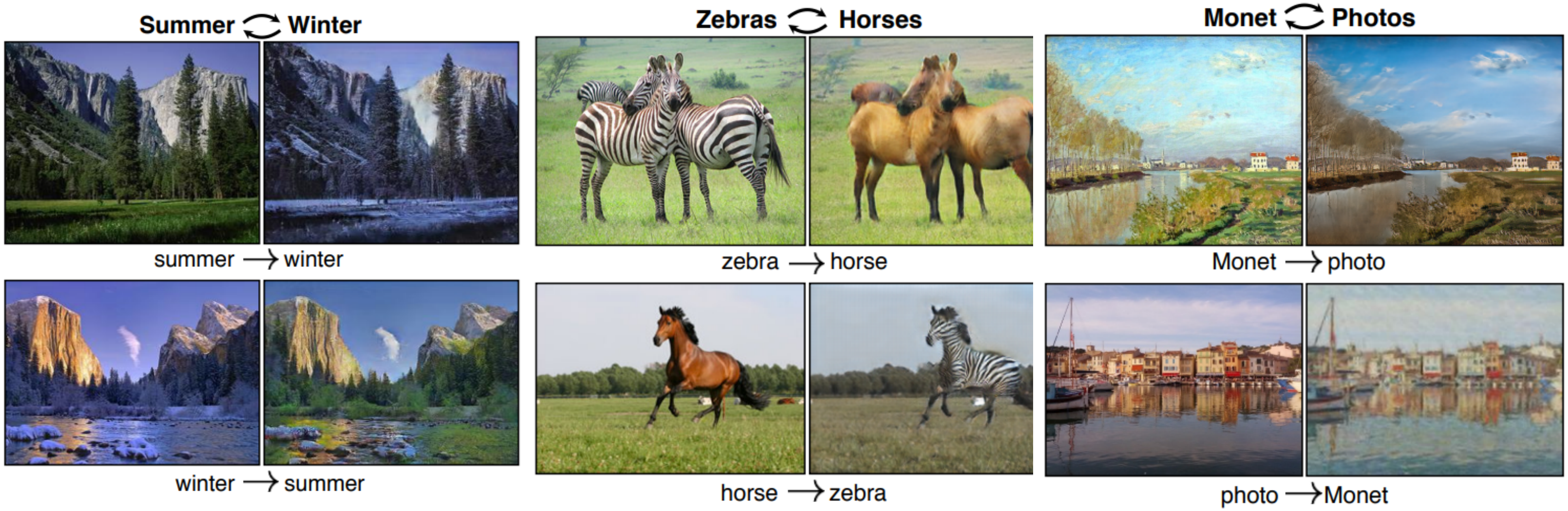
$$\mathcal{L}_{cyc}(G_1, G_2) = \mathbb{E}\left[\left\|G_2(G_1(x)) - x\right\|_1\right] + \left[\left\|G_1(G_2(y)) - y\right\|_1\right]$$



Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

CycleGAN

- Example results

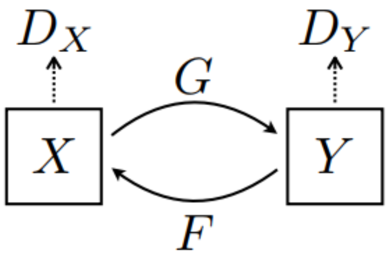


Project Page: https://junyanz.github.io/CycleGAN/

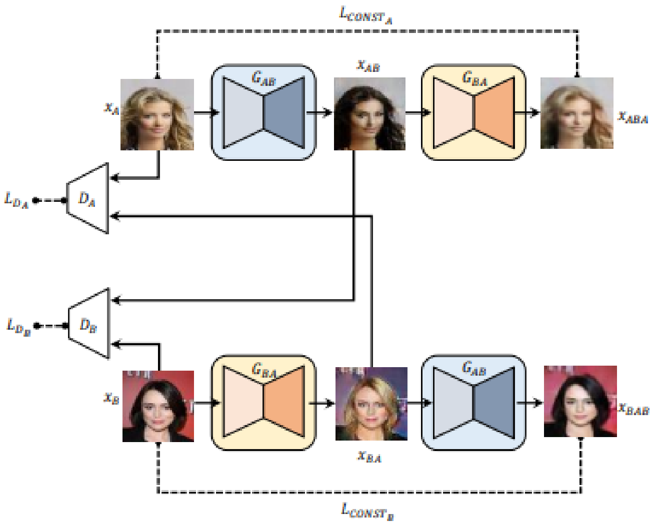
Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

Image Translation Using Unpaired Training Data

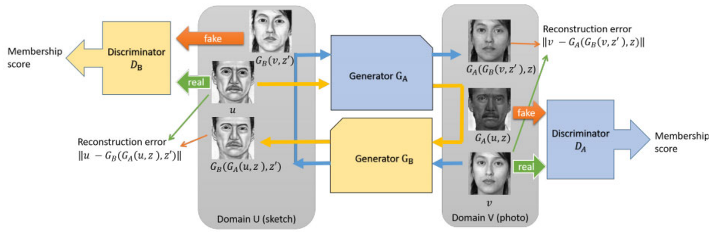
- CycleGAN, DiscoGAN, and DualGAN



CycleGAN
ICCV'17



DiscoGAN
ICML'17



DualGAN
ICCV'17

Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *ICCV* 2017.
Kim et al. "Learning to Discover Cross-Domain Relations with Generative Adversarial Networks.", *ICML 2017*
Yi, Zili, et al. "Dualgan: Unsupervised dual learning for image-to-image translation." *ICCV 2017*

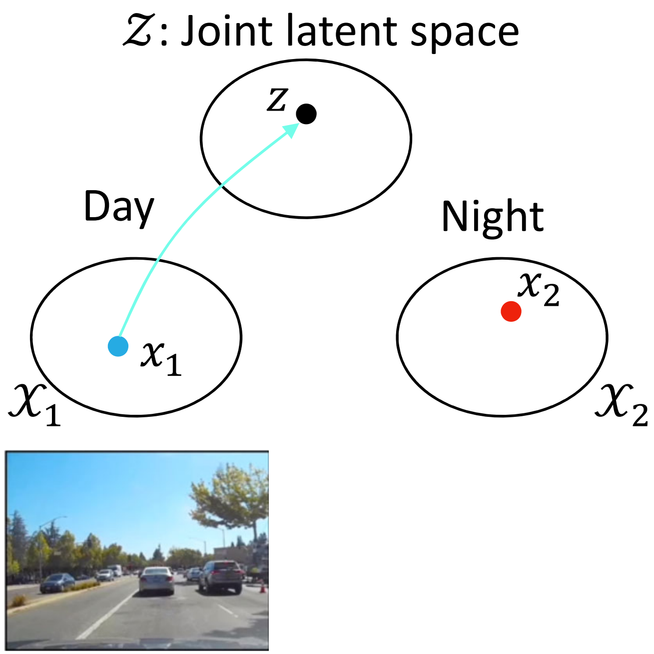
Transfer Learning for Image Synthesis

- Cross-Domain Image Translation
  - Pix2pix: Pairwise cross-domain training data
  - CycleGAN/DualGAN/DiscoGAN: Unpaired cross-domain training data
  - UNIT: Learning cross-domain image representation (with unpaired training data)
  - AdaIN
  - Beyond image translation

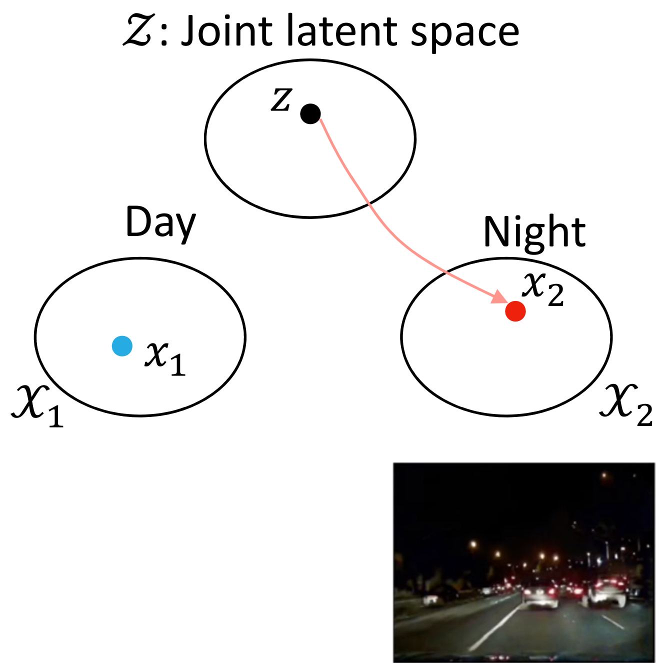
UNIT

- Unsupervised Image-to-Image Translation Networks (NIPS’17)
  - Image translation via learning cross-domain joint representation

Stage1: Encode to the joint space



Stage2: Generate cross-domain images



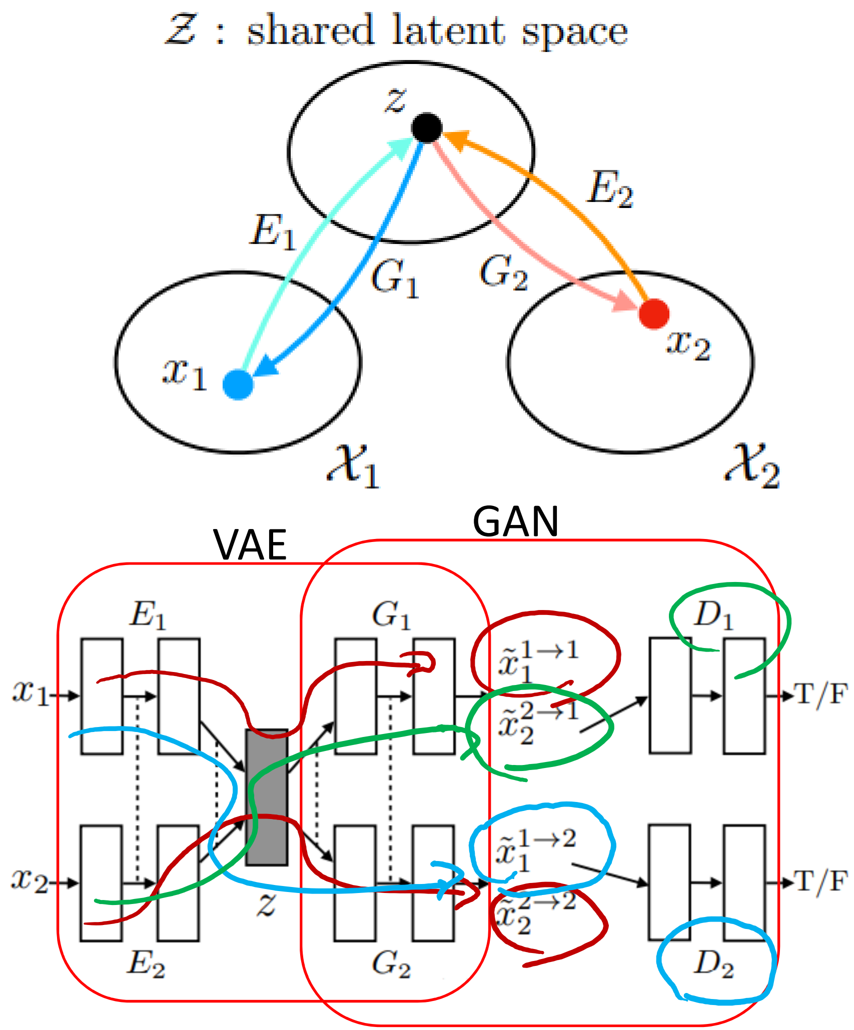
Liu et al., "Unsupervised image-to-image translation networks.“, NIPS 2017

UNIT

- Goal/Problem Setting
  - Image translation
    across two distinct domains
  - Unpaired training image data
- Idea
  - Based on two parallel VAE-GAN models

source

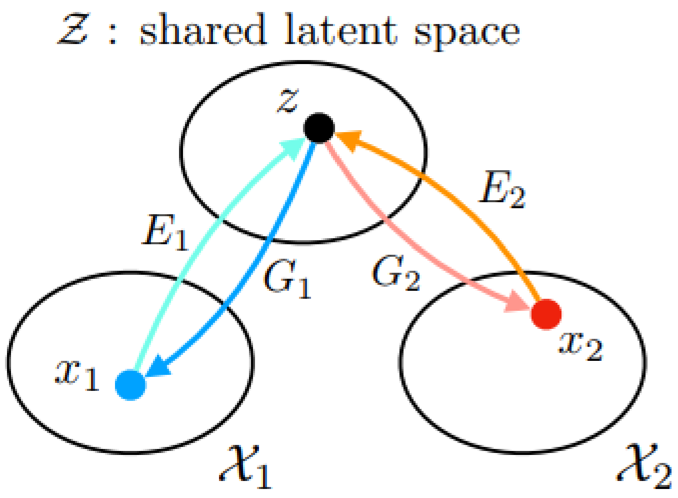
tgt

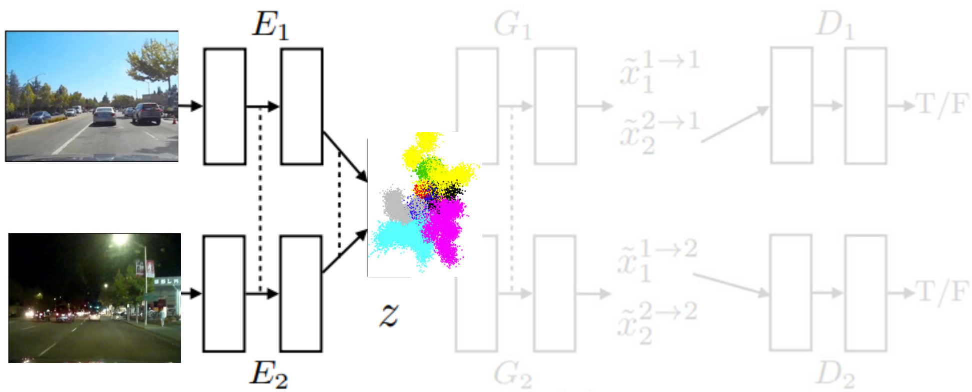


Liu et al., "Unsupervised image-to-image translation networks.", NIPS 2017

UNIT

- Goal/Problem Setting
  - Image translation
    across two distinct domains
  - Unpaired training image data
- Idea
  - Based on two parallel VAE-GAN models
  - Learning of joint representation
    across image domains

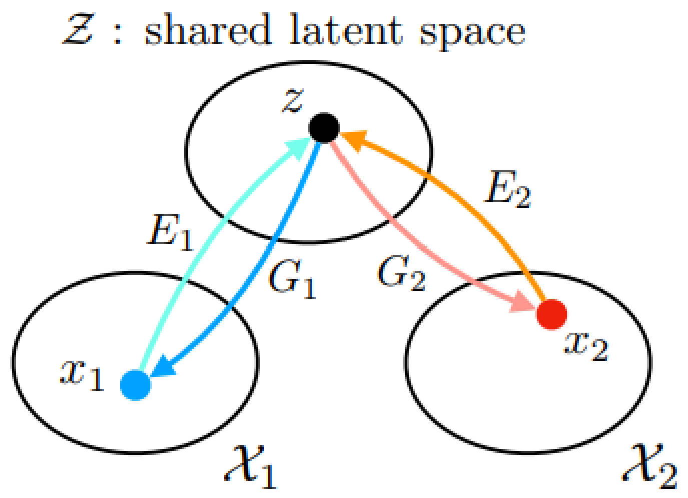


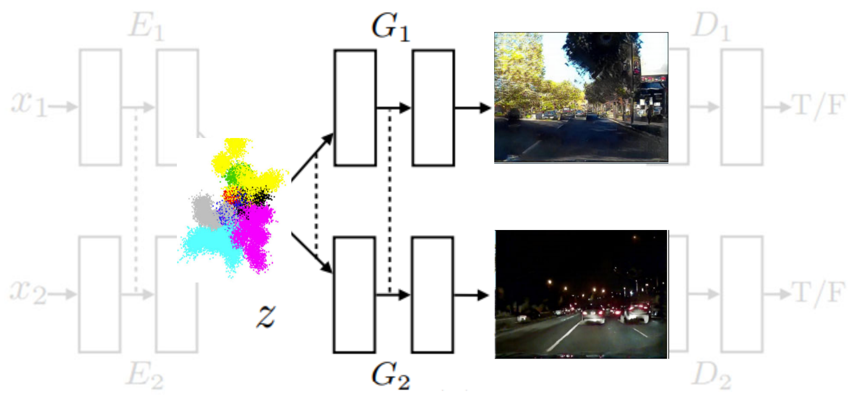


Liu et al., "Unsupervised image-to-image translation networks.", NIPS 2017

UNIT

- Goal/Problem Setting
  - Image translation
    across two distinct domains
  - Unpaired training image data
- Idea
  - Based on two parallel VAE-GAN models
  - Learning of joint representation
    across image domains
  - Generate cross-domain images
    from joint representation





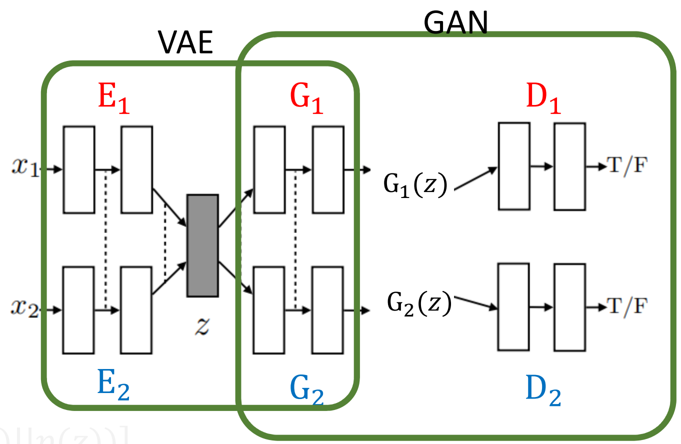
Liu et al., "Unsupervised image-to-image translation networks.", NIPS 2017

UNIT

- ***Learning***

Overall objective function

$$G^* = \arg\min_{G}\max_{D} \underbrace{\mathcal{L}_{VAE}(E_1, G_1, E_2, G_2)}_{\textbf{Variation Autoencoder}} + \underbrace{\mathcal{L}_{GAN}(G_1, D_1, G_2, D_2)}_{\textbf{Adversarial}}$$



Variation Autoencoder Loss

$$\mathcal{L}_{VAE}(E_1, G_1, E_2, G_2) = \mathbb{E}[||G_1(E_1(x_1)) - x_1||_2] + \mathbb{E}[\mathcal{KL}(q_1(z)||p(z))]$$
$$\mathbb{E}[||G_2(E_2(x_2)) - x_2||_2] + \mathbb{E}[\mathcal{KL}(q_2(z)||p(z))]$$

Adversarial Loss

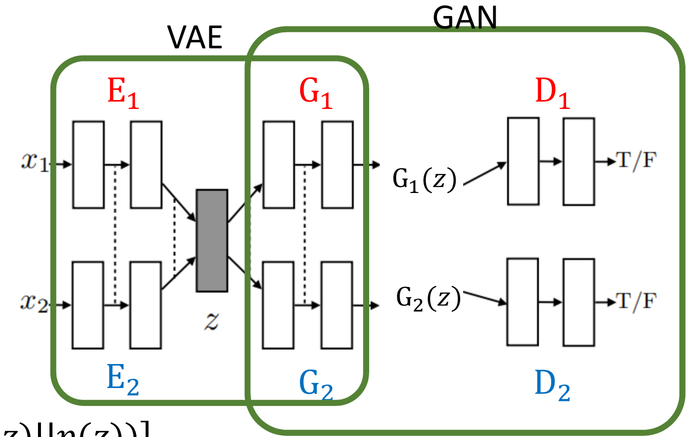
$$\mathcal{L}_{GAN}(G_1, D_1, G_2, D_2) = \mathbb{E}[\log(1 - D_1(G_1(z))] + \mathbb{E}[\log D_1(y_1)]$$
$$\mathbb{E}[\log(1 - D_2(G_2(z))] + \mathbb{E}[\log D_2(y_2)]$$

UNIT

- *Learning*

Overall objective function

$$G^* = \arg\min_{G}\max_{D} \underbrace{\mathcal{L}_{VAE}(E_1, G_1, E_2, G_2)}_{\textbf{Variation Autoencoder}} + \underbrace{\mathcal{L}_{GAN}(G_1, D_1, G_2, D_2)}_{\textbf{Adversarial}}$$



Variation Autoencoder Loss

$$\mathcal{L}_{VAE}(E_1, G_1, E_2, G_2) = \mathbb{E}[\|G_1(E_1(x_1)) - x_1\|_2] + \mathbb{E}[\mathcal{KL}(q_1(z)||p(z))]$$
$$\mathbb{E}[\|G_2(E_2(x_2)) - x_2\|_2] + \mathbb{E}[\mathcal{KL}(q_2(z)||p(z))]$$

Adversarial Loss

$$\mathcal{L}_{GAN}(G_1, D_1, G_2, D_2) = \mathbb{E}[log(1 - D_1(G_1(z))] + \mathbb{E}[log\, D_1(y_1)]$$
$$\mathbb{E}[log(1 - D_2(G_2(z))] + \mathbb{E}[log\, D_2(y_2)]$$

Generated Real

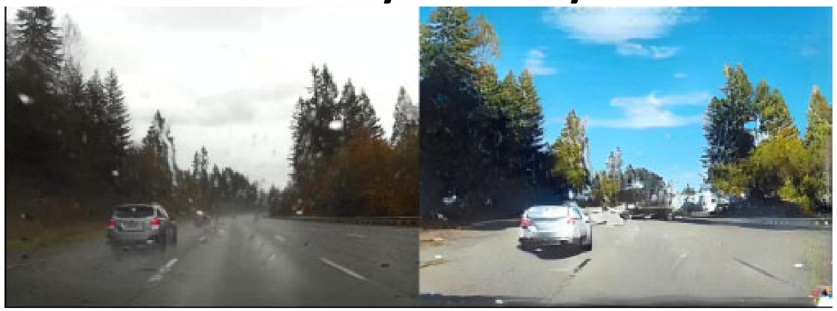
UNIT

- Example results

Sunny → Rainy

Real Street-view → Synthetic Street-view

Rainy → Sunny



Synthetic Street-view → Real Street-view

Github Page: https://github.com/mingyuliutw/UNIT

Liu et al., "Unsupervised image-to-image translation networks.“, NIPS 2017

Transfer Learning for Image Synthesis

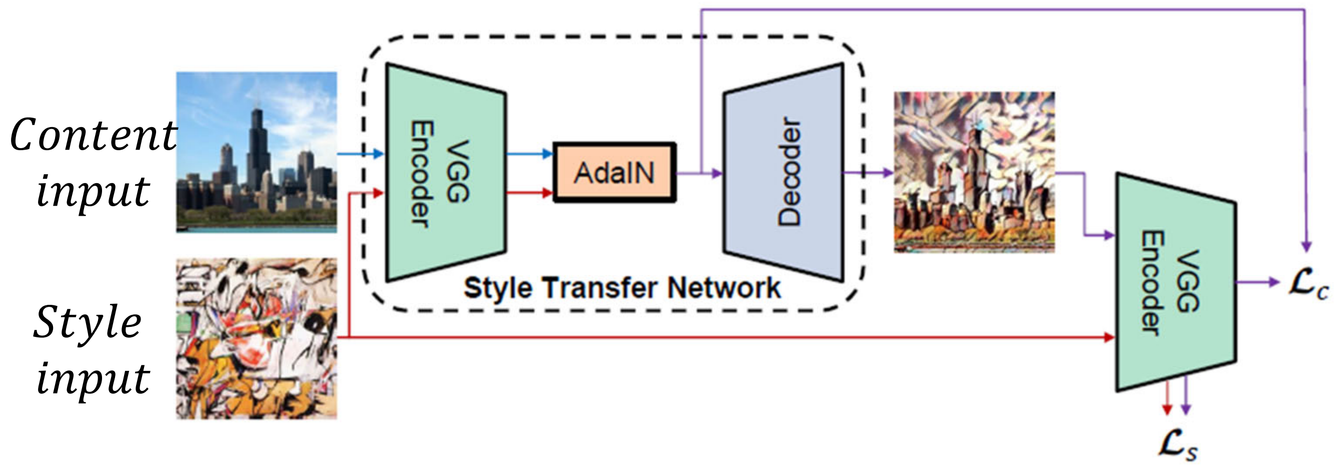
- Cross-Domain Image Translation
  - Pix2pix: Pairwise cross-domain training data
  - CycleGAN/DualGAN/DiscoGAN: Unpaired cross-domain training data
  - UNIT: Learning cross-domain image representation (with unpaired training data)
  - AdaIN: Single-image arbitrary style transfer in real-time
  - Beyond image translation

AdaIN

- We've talked about style transfer methods like Pix2Pix or CycleGAN.
- Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization (ICCV'17)
  - Single-image arbitrary style transfer in real-time

Huang *et al.* " Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization . " *ICCV* 2017

AdaIN



- Adaptive Instance Normalization

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

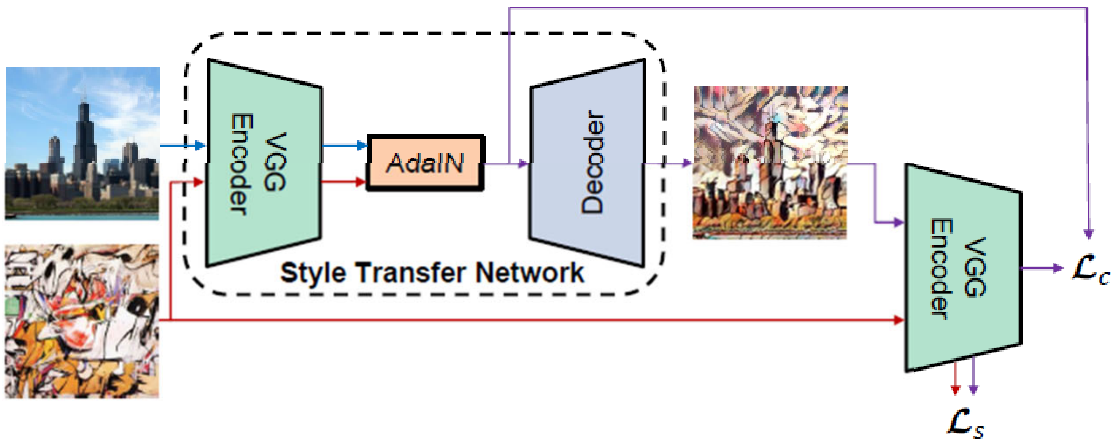
- x: content input, y: style input
- No learnable affine parameters
- Perform style transfer in the feature space

Huang *et al.* " Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization . " *ICCV* 2017

AdaIN

- f: Encoder, g: Decoder

$$t = \mathrm{AdaIN}(f(c), f(s))$$



$$\mathcal{L}_c = \|f(g(t)) - t\|_2 \quad \textit{Content loss: content consistency}$$

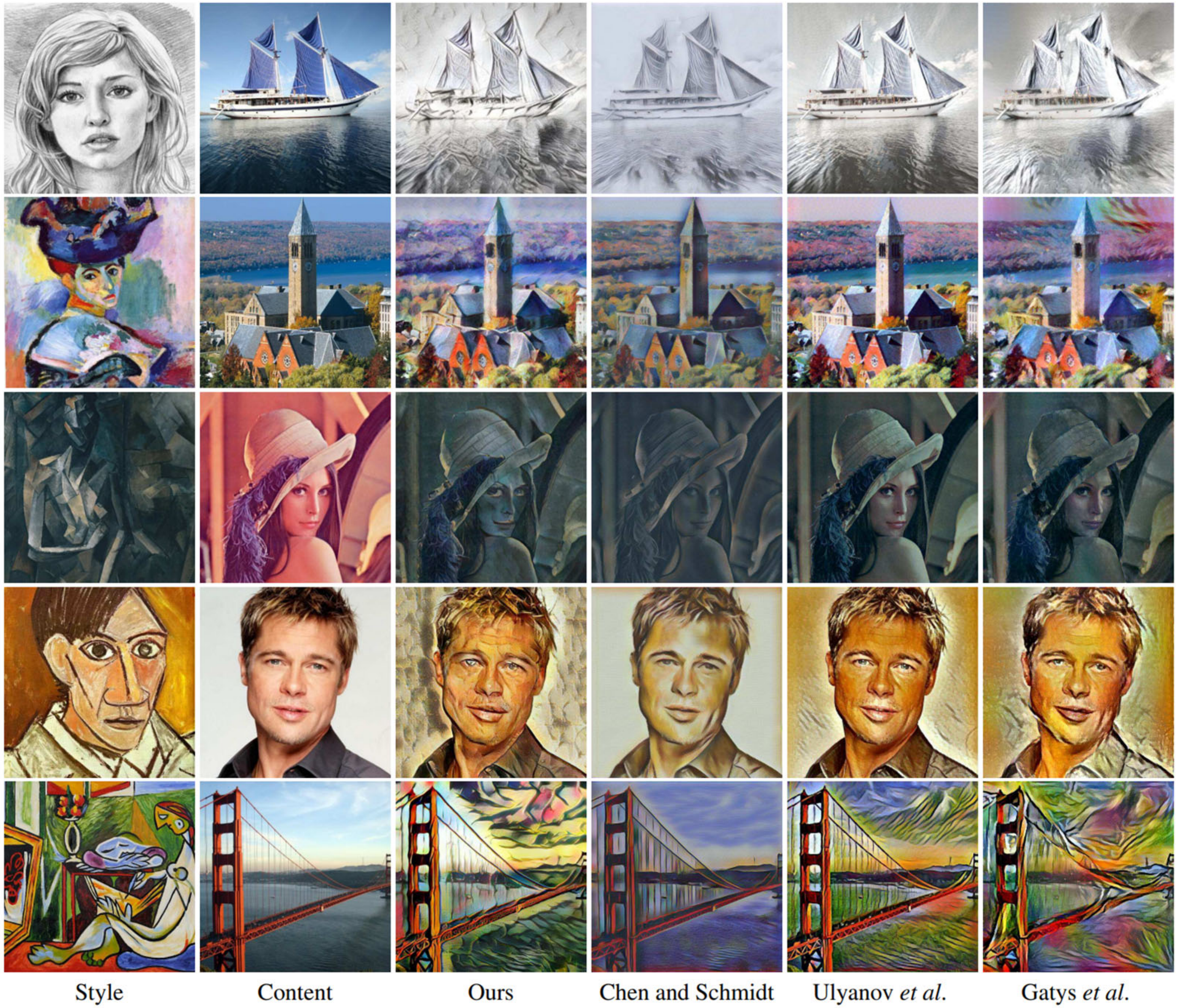
$$\mathcal{L}_s = \sum_{i=1}^{L} \|\mu(\phi_i(g(t))) - \mu(\phi_i(s))\|_2 + \sum_{i=1}^{L} \|\sigma(\phi_i(g(t))) - \sigma(\phi_i(s))\|_2$$

$Style\ loss: Gram\ matrix\ loss$
$(\emptyset_i\ denotes\ a\ layer\ in\ VGG - 19\ used\ to\ compute\ style\ loss)$

Huang *et al.* "Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization ." *ICCV* 2017

AdaIN

- Qualitative results



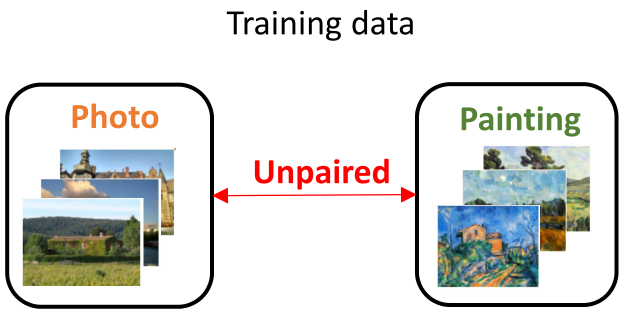
Huang *et al.* " Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization . " *ICCV* 2017

Transfer Learning for Image Synthesis

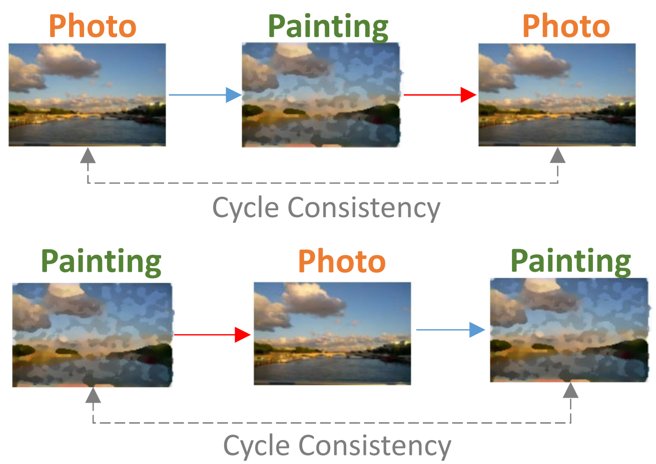
- Cross-Domain Image Translation
  - Pix2pix: Pairwise cross-domain training data
  - CycleGAN/DualGAN/DiscoGAN: Unpaired cross-domain training data
  - UNIT: Learning cross-domain image representation (with unpaired training data)
  - AdaIN
  - Beyond image translation

Revisit: CycleGAN

- **Goal / Problem Setting**
  - Image translation across two distinct domains
  - Unpaired training data

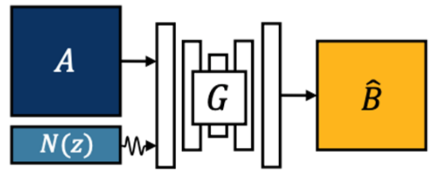


- **Idea**
  - Autoencoding-like image translation
  - Cycle consistency between two domains



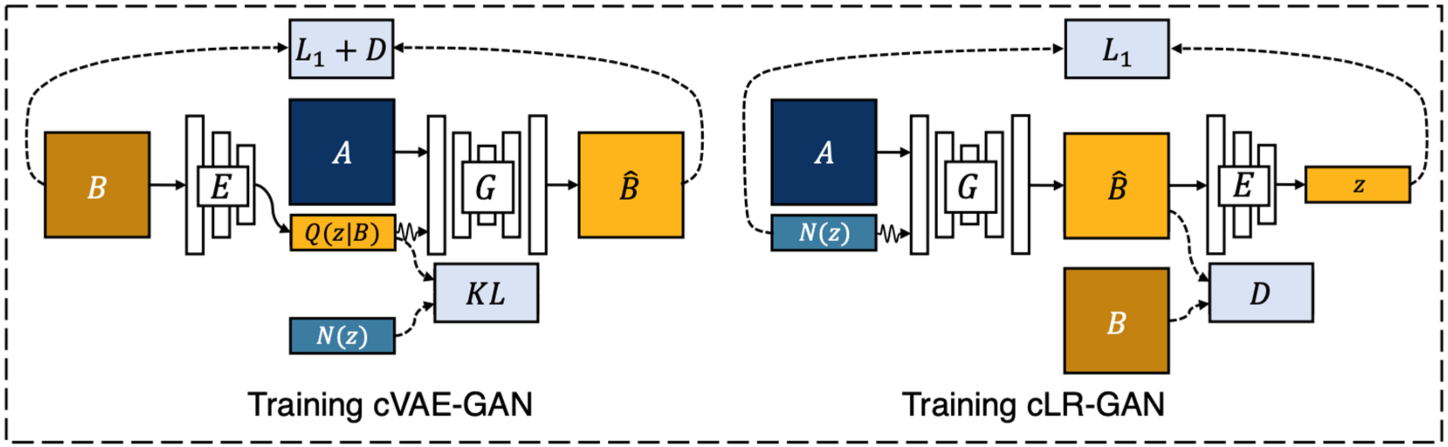
Zhu et al. "Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks." *CVPR* 2017.

BicycleGAN



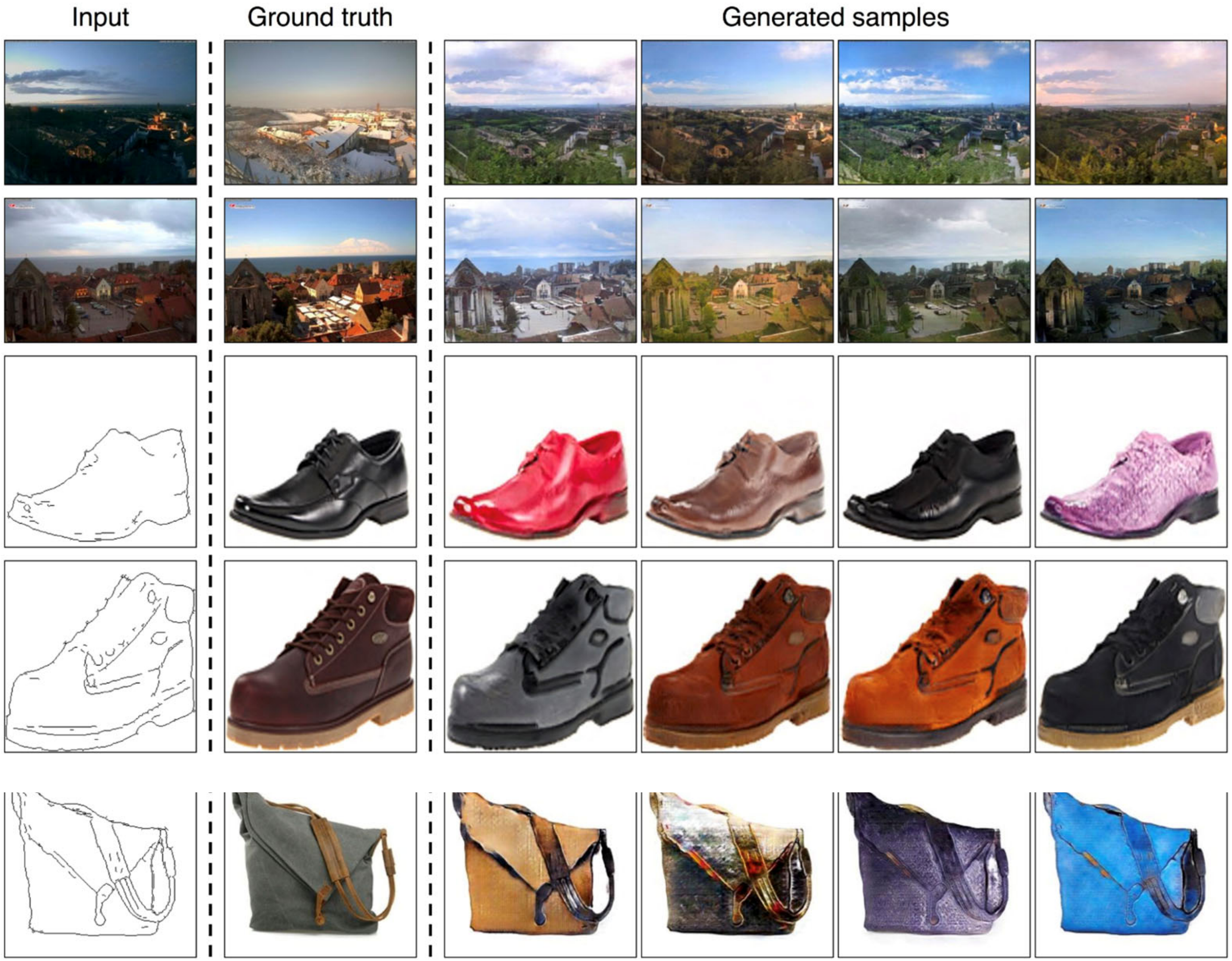
(a) Testing Usage for all models

- **Toward Multimodal Image-to-Image Translation** (NIPS'17)
- **Goal / Problem Setting**
  - Producing diverse images across two distinct domains.
  - Pairwise training data
- **Idea**
  - Combine conditional VAE-GAN and conditional Latent Regressor GAN.



Training BicycleGAN

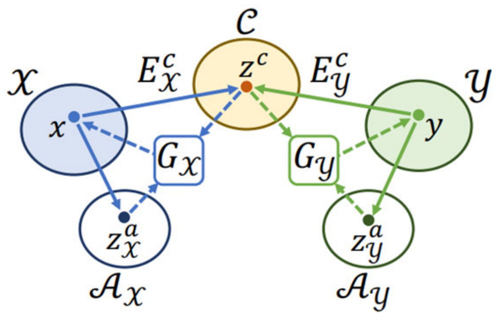
BicycleGAN - Experiment

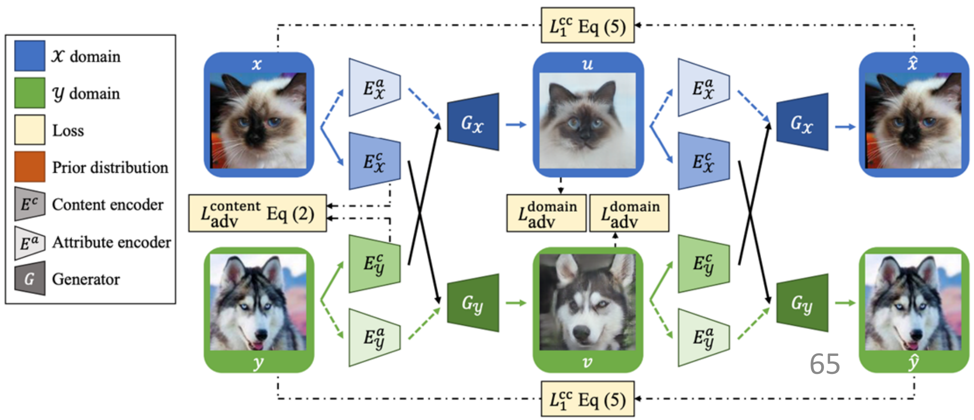


Zhu *et al.*, Toward Multimodal Image-to-Image Translation, NIPS 2017

DRIT

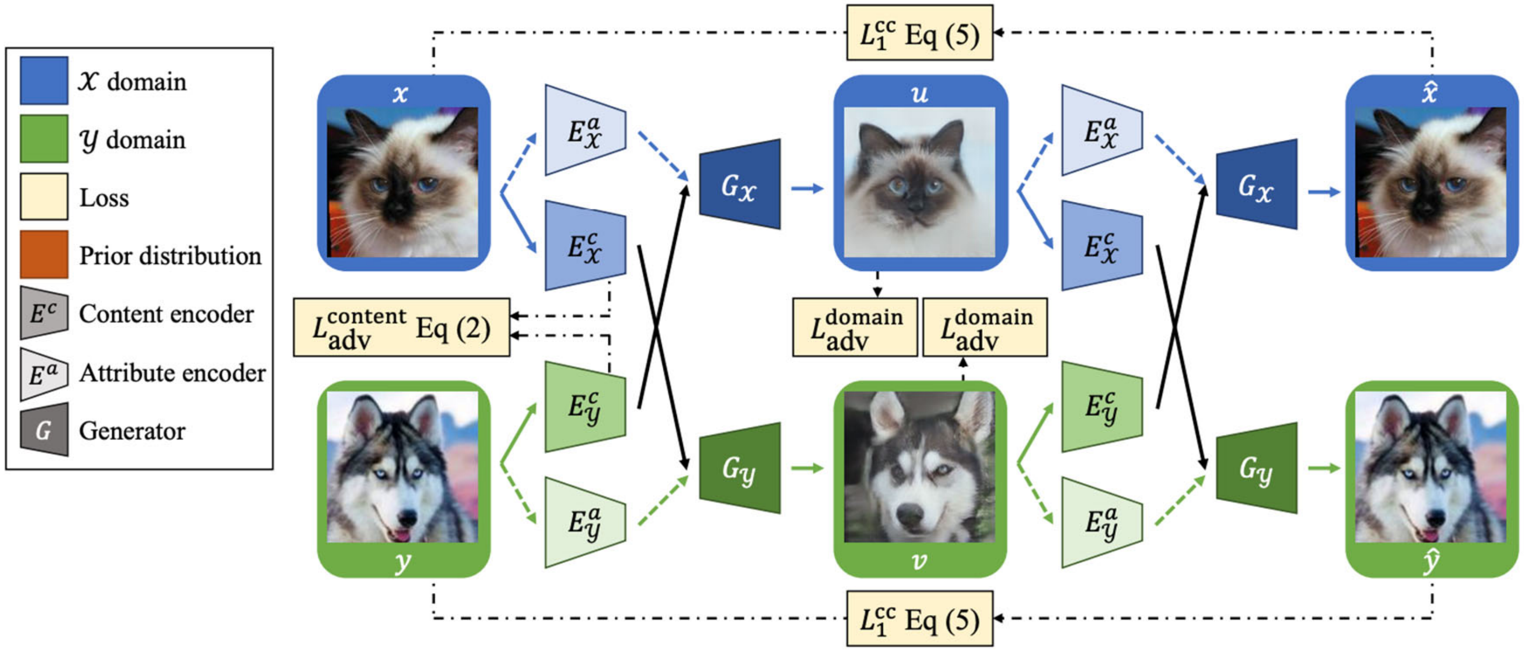
- **Diverse Image-to-Image Translation via Disentangled Representations** (ECCV'18 oral)
- **Goal / Problem Setting**
  - Producing diverse images across two distinct domains.
  - Unpaired training data
- **Idea**
  - Disentangle latent representation into domain-invariant and domain-specific features
  - Generate cross-domain images by swapping the latent feature from each domain.
  - Applied cross-cycle consistency





Method – Main Framework

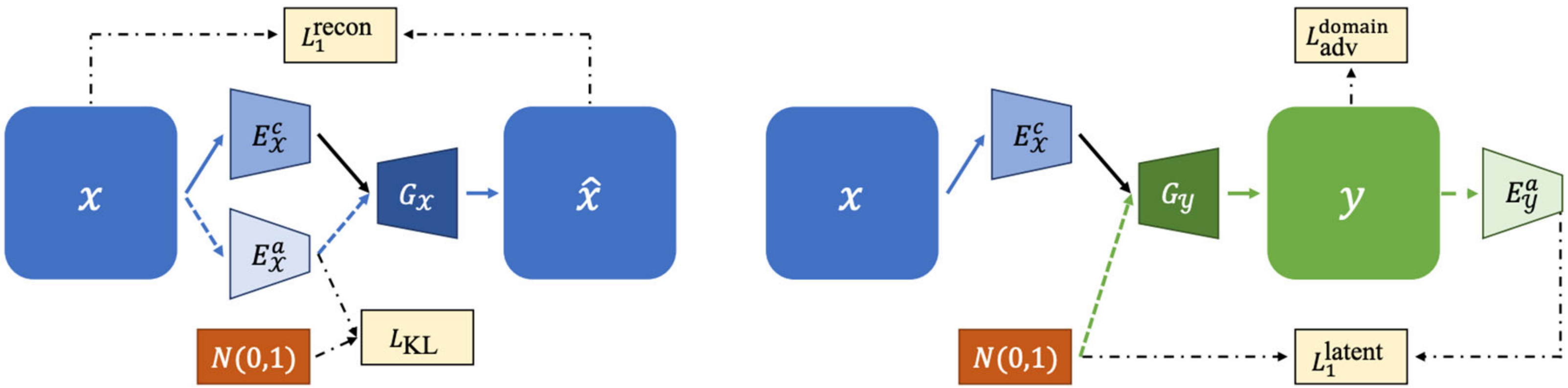
Attribute: species
Content: pose (style)



Lee *et al.*, Diverse Image-to-Image Translation via Disentangled Representations, ECCV 2018 (oral)

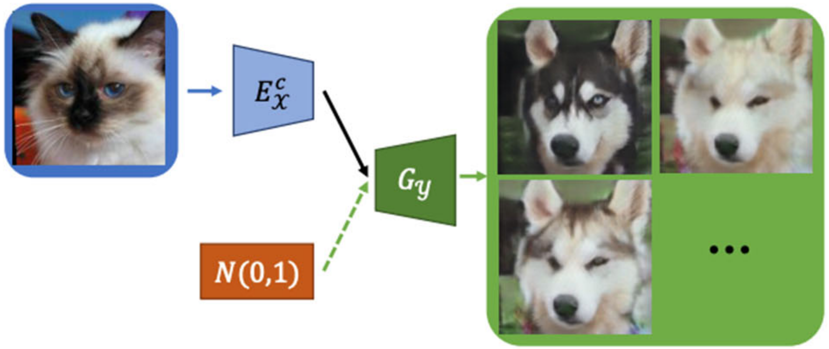
Method – For Attribute Features

- **KL loss**:
  perform stochastic sampling at test time.
- **Latent regression loss**:
  encourage invertible mapping btw image and latent representations

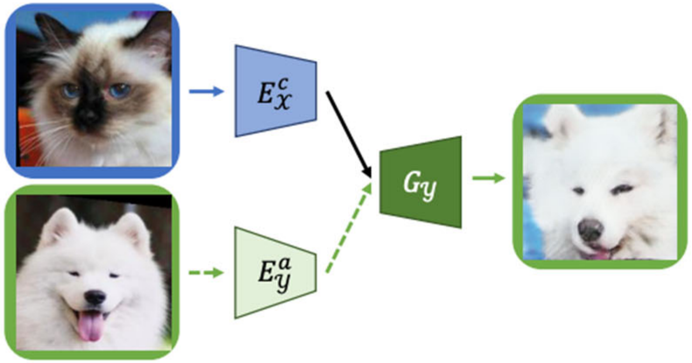


Lee *et al*., Diverse Image-to-Image Translation via Disentangled Representations, ECCV 2018 (oral)

Method – Inference phase



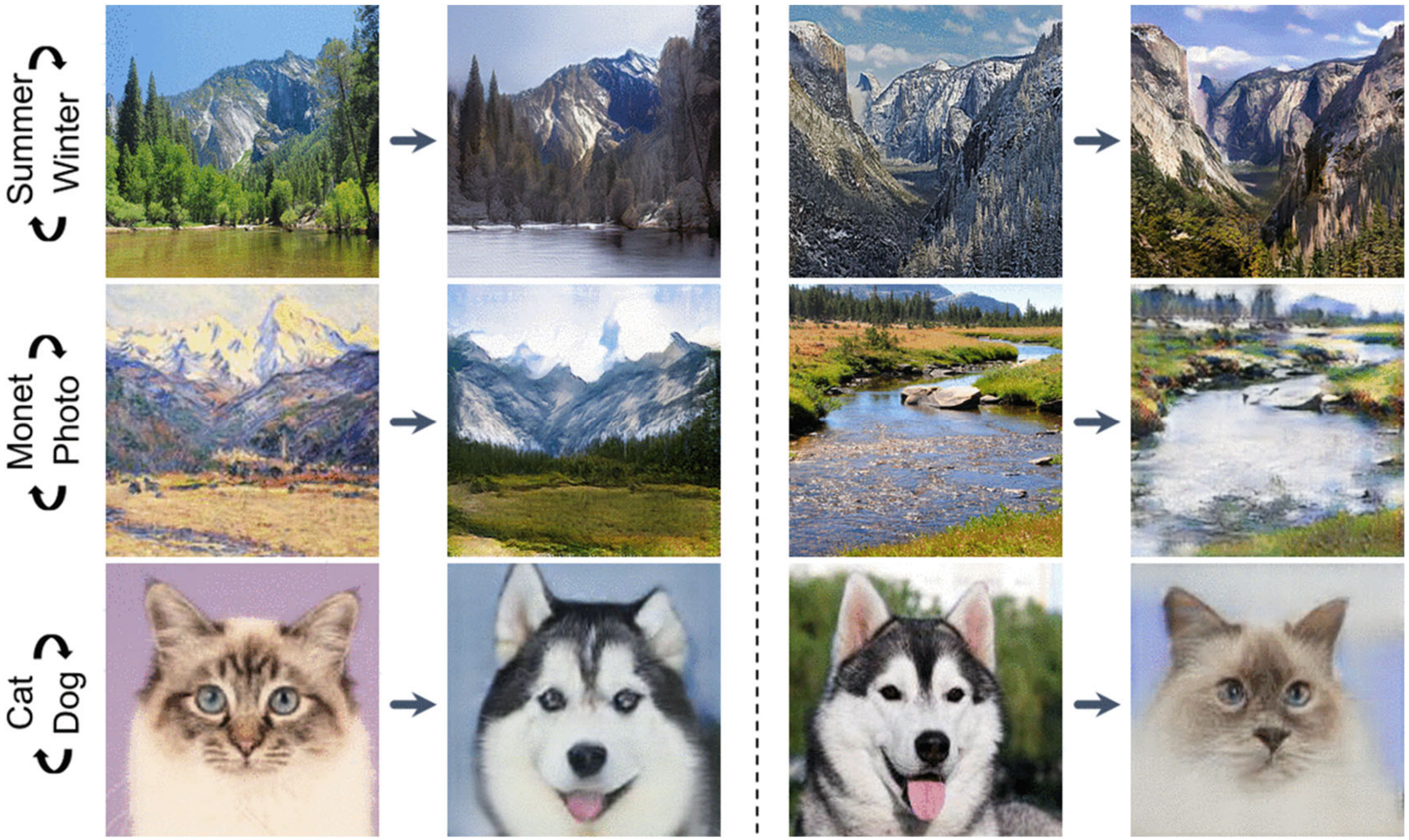
(b) Testing with random attributes



(c) Testing with a given attribute

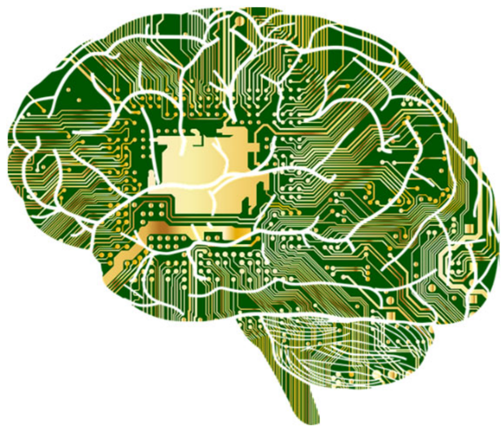
Lee *et al*., Diverse Image-to-Image Translation via Disentangled Representations, ECCV 2018 (oral)

Example Results



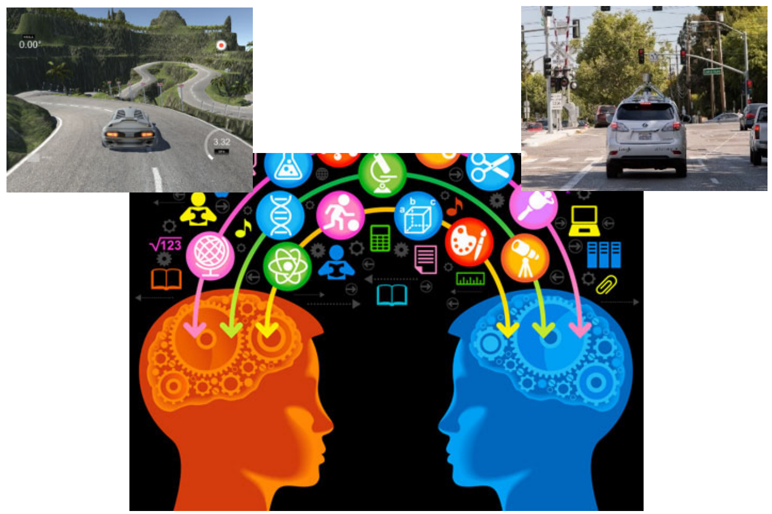
Lee *et al*., Diverse Image-to-Image Translation via Disentangled Representations, ECCV 2018 (oral)

What to Be Covered Today…



- Generative Models
  - Diffusion Model
- Transfer Learning
  - Visual Classification – Domain Adaptation
  - Visual Synthesis – Style Transfer
- Representation Disentanglement
  - Supervised vs. unsupervised feature disentanglement

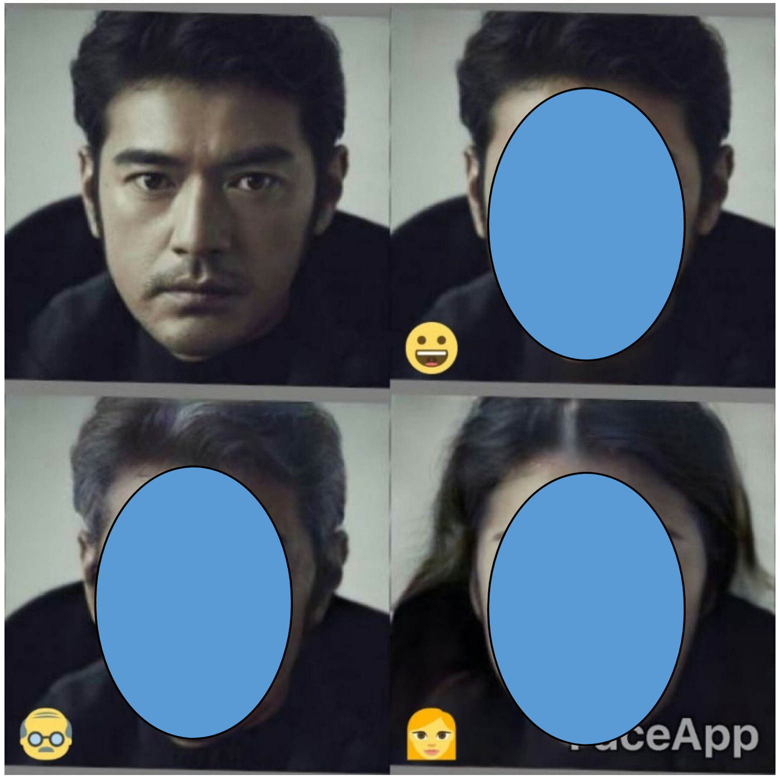
Sprouts in the shape of text 'Imagen' coming out of a fairytale book.



Beyond Image Style Transfer: Learning Interpretable Deep Representations

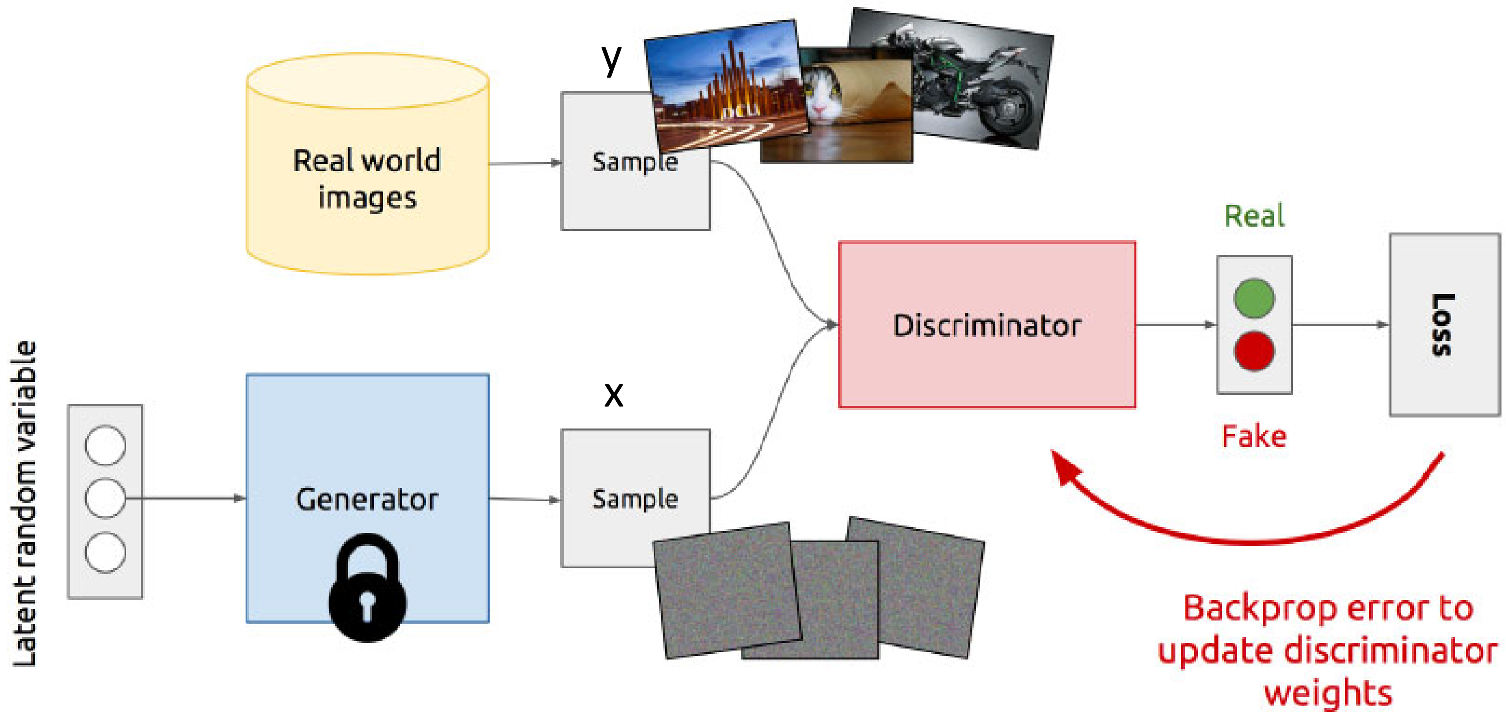
- Faceapp – Putting a smile on your face!
  - Deep learning for representation disentanglement
  - Interpretable deep feature representation

Input
Mr. Takeshi Kaneshiro



Recall: Generative Adversarial Networks (GAN)

- Architecture of GAN
  - Loss $\mathcal{L}_{GAN}(G,D) = \mathbb{E}[\log(1 - D\ (G(x)))] + \mathbb{E}[\log D(y)]$



Goodfellow et al., Generative Adversarial Nets, NIPS, 2014

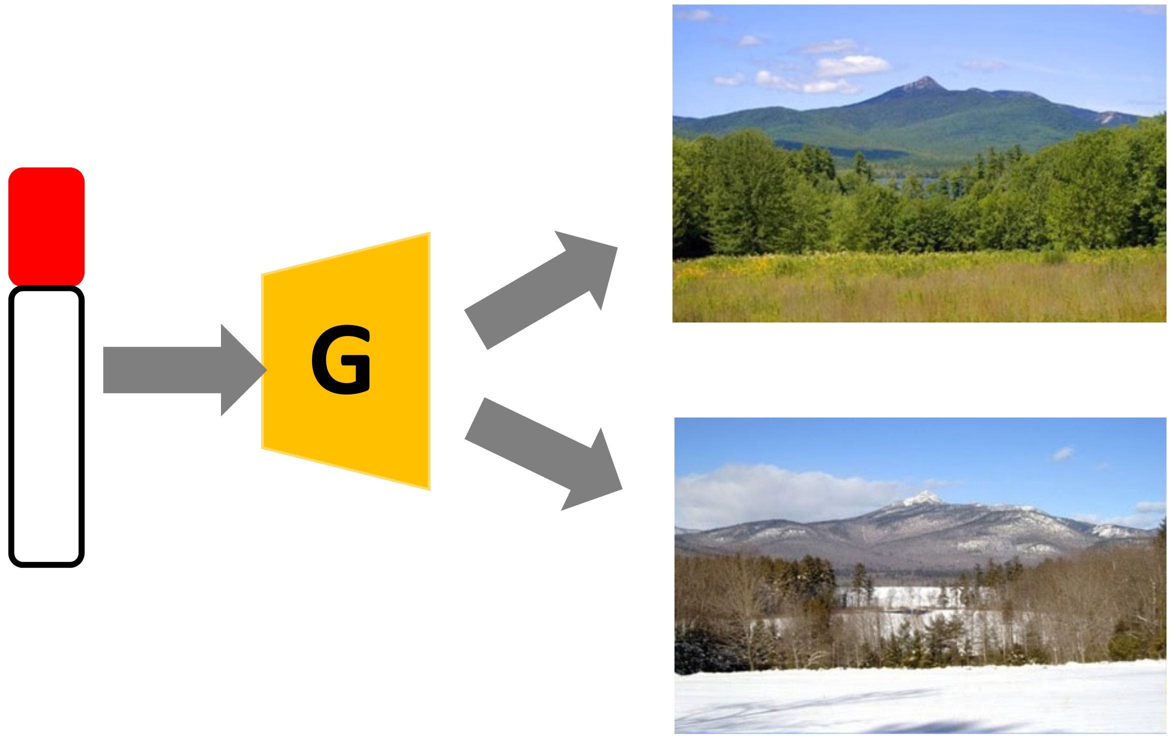
Representation Disentanglement

- Goal
  - Interpretable deep feature representation
  - Disentangle attribute of interest ***c*** from the derived latent representation ***z***
  - Possible solutions: VAE, GAN, or mix of them…

(label, attribute, etc.)
condition

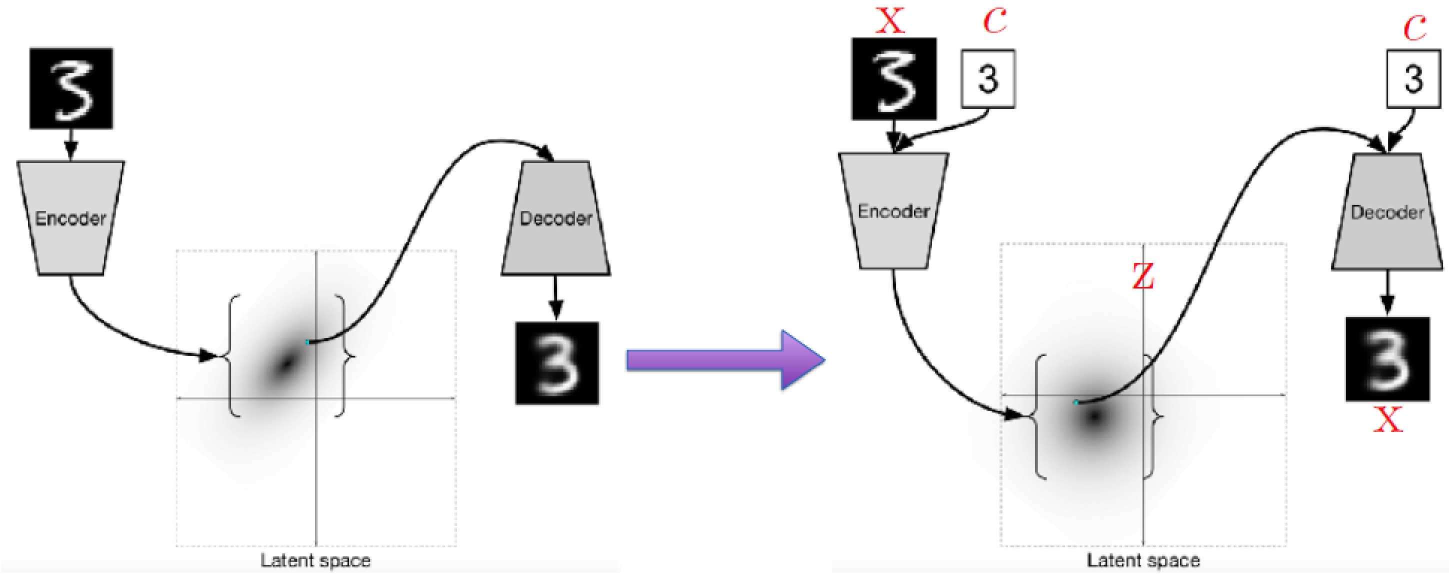
Interpretable Factor *c*
(e.g., season)

Latent feature *z*
(uninterpretable)



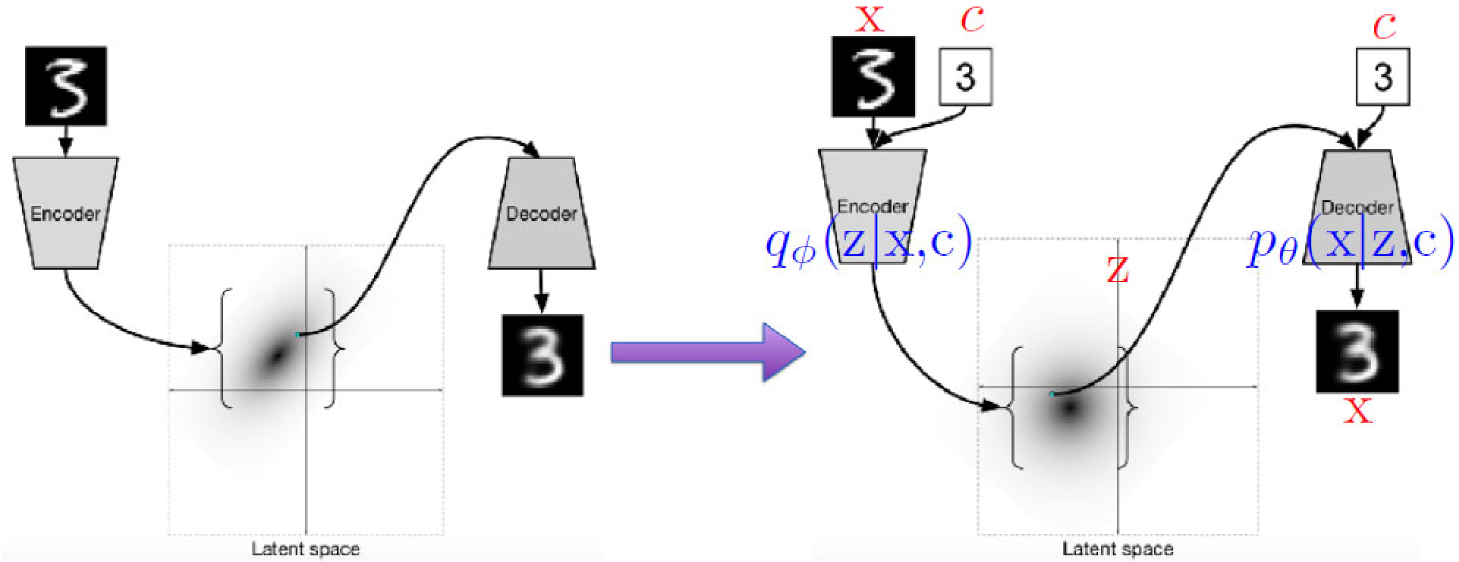
Representation Disentanglement

- Goal
  - Interpretable deep feature representation
  - Disentangle attribute of interest ***c*** from the derived latent representation ***z***
    - Supervised setting: from VAE to conditional VAE



Representation Disentanglement

- Conditional VAE
  - Given training data ***x*** and attribute of interest c, we model the conditional distribution $p_\theta(x|c)$.

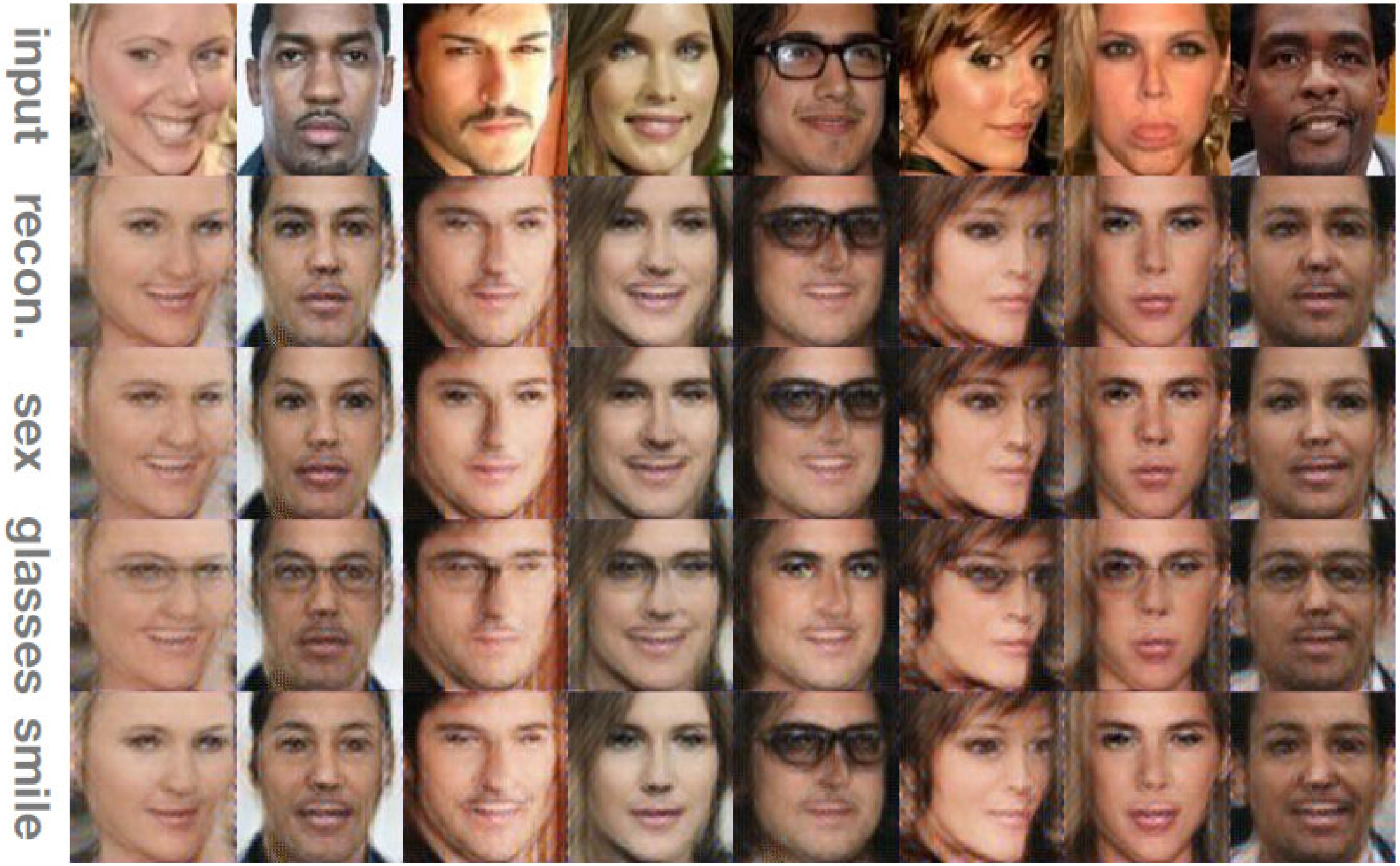


$$= \underbrace{-KL(q_\phi(z|x,c)||p_\theta(z|c))}_{\text{impose prior}} + \underbrace{\mathbb{E}_{q_\phi(z|x,c)} \log p_\theta(x|c,z)}_{\text{reconstruction}}$$

https://zhuanlan.zhihu.com/p/25518643

Representation Disentanglement

- Conditional VAE
  - Example Results



Representation Disentanglement

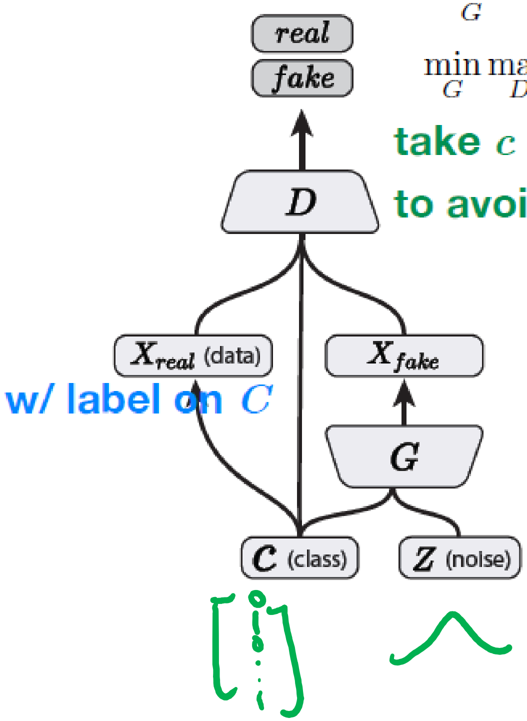
- Conditional GAN
  - Interpretable latent factor ***c***
  - Latent representation ***z***

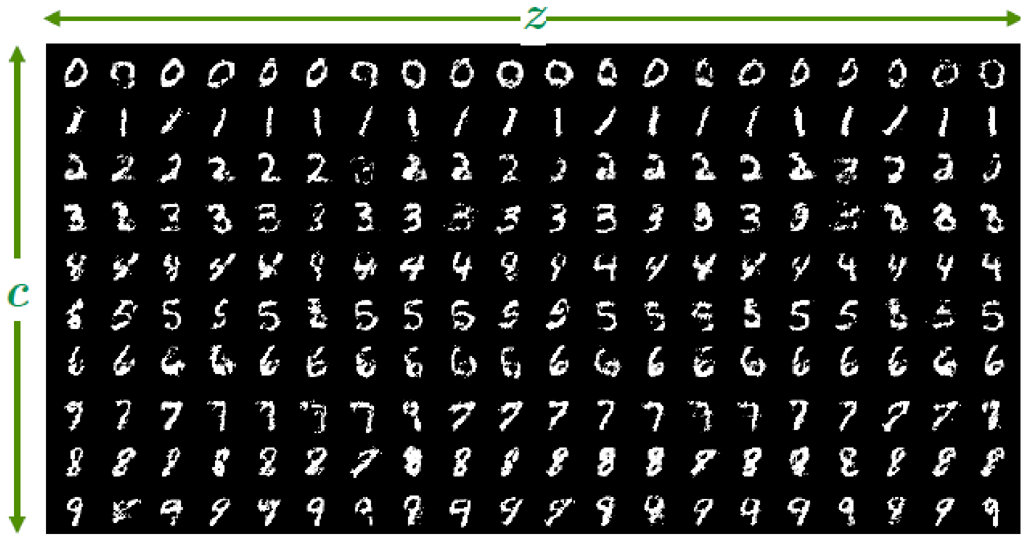
$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

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

take c as also input for D

to avoid ignoring c in G

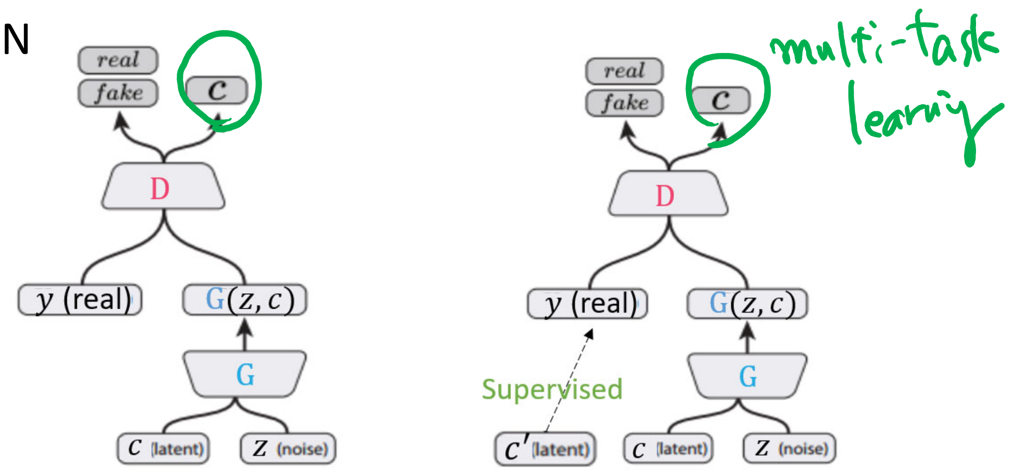




https://arxiv.org/abs/1411.1784

Representation Disentanglement

- Goal
  - Interpretable deep feature representation
  - Disentangle attribute of interest ***c*** from the derived latent representation ***z***
    - Unsupervised: InfoGAN
    - Supervised: AC-GAN



InfoGAN

Chen et al.
NIPS '16

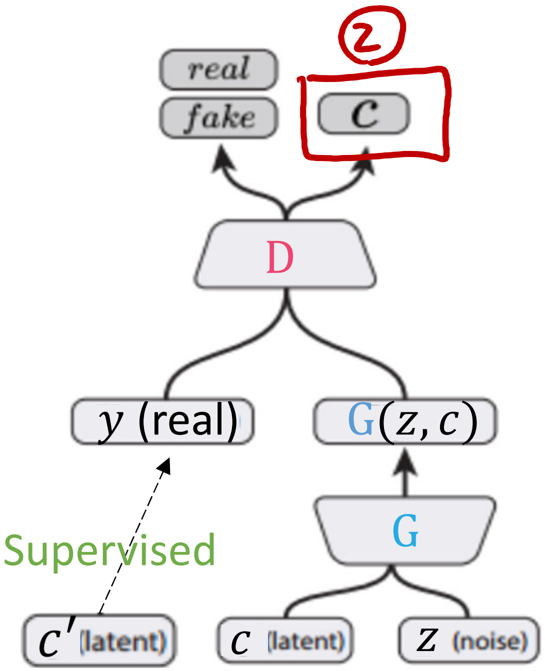
ACGAN

Odena et al.
ICML '17

Chen et al., InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets., NIPS 2016.
Odena et al., Conditional image synthesis with auxiliary classifier GANs. ICML'17

AC-GAN

- Supervised Disentanglement



- **Learning**
  - **Overall objective function**

$$G^* = \arg\min_{G}\max_{D} \underbrace{\mathcal{L}_{GAN}(G, D)} + \underbrace{\mathcal{L}_{cls}(G, D)}_{2}$$

  - **Adversarial Loss**

$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}[\log(1 - D(G(z, c)))] + \mathbb{E}[\log D(y)]$$

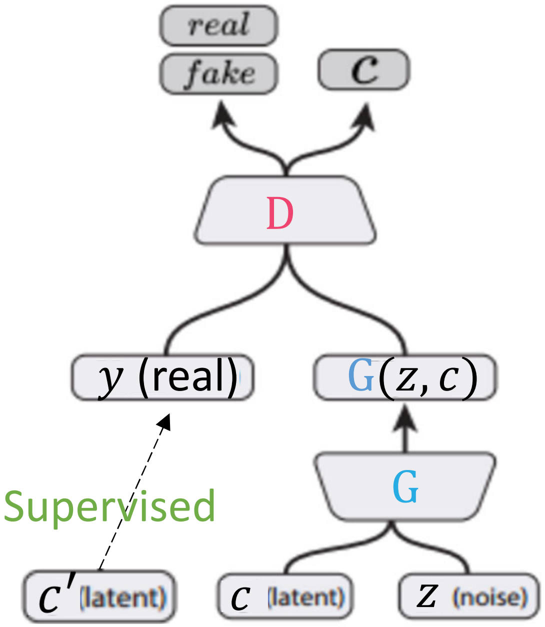
  - **Disentanglement loss**

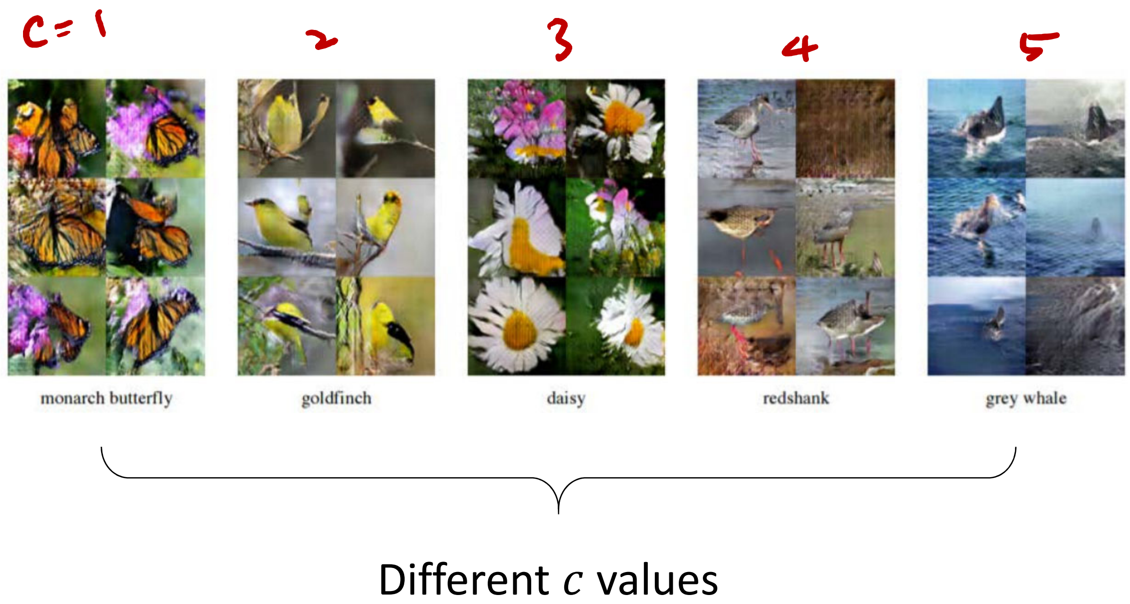
$$\mathcal{L}_{cls}(G, D) = \underbrace{\mathbb{E}[-\log D_{cls}(c'|y)]}_{\text{Real data w.r.t. its domain label}} + \underbrace{\mathbb{E}[-\log D_{cls}(c|G(x, c))]}_{\text{Generated data w.r.t. assigned label}}$$

Odena et al., Conditional image synthesis with auxiliary classifier GANs. ICML'17

AC-GAN

- Supervised Disentanglement

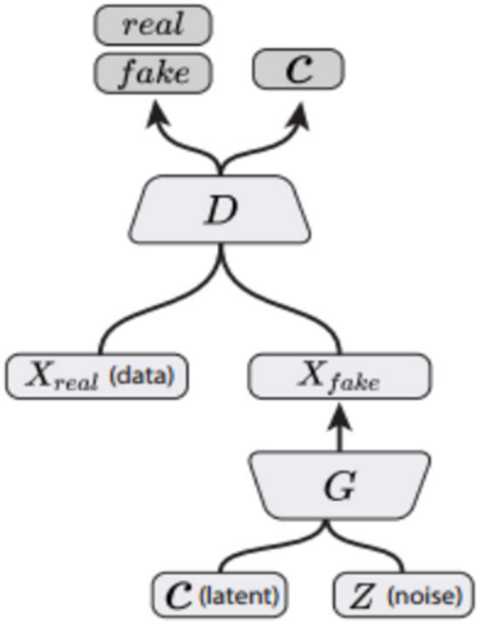




Odena et al., Conditional image synthesis with auxiliary classifier GANs. ICML'17

InfoGAN

- Unsupervised Disentanglement



- Learning
  - **Overall objective function**

$$G^* = \arg\min_{G}\max_{D} \mathcal{L}_{GAN}(G, D) + \mathcal{L}_{cls}(G, D)$$

  - **Adversarial Loss**

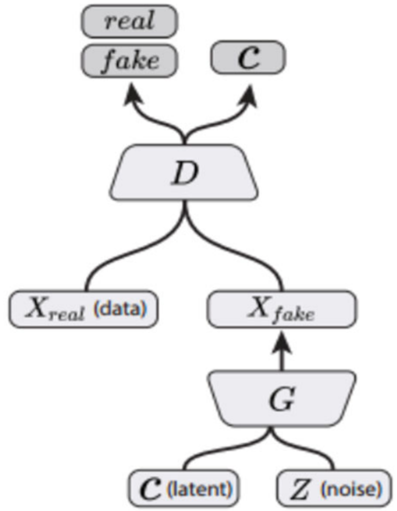
$$\mathcal{L}_{GAN}(G, D) = \mathbb{E}[\log(1 - D(G(z, c)))] + \mathbb{E}[\log D(y)]$$

  - **Disentanglement loss**

$$\mathcal{L}_{cls}(G, D) = \underbrace{\mathbb{E}[-\log D_{cls}(c|G(x, c))]}_{\text{Generated data w.r.t. assigned label}}$$

Chen et al., InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets., NIPS 2016.

InfoGAN



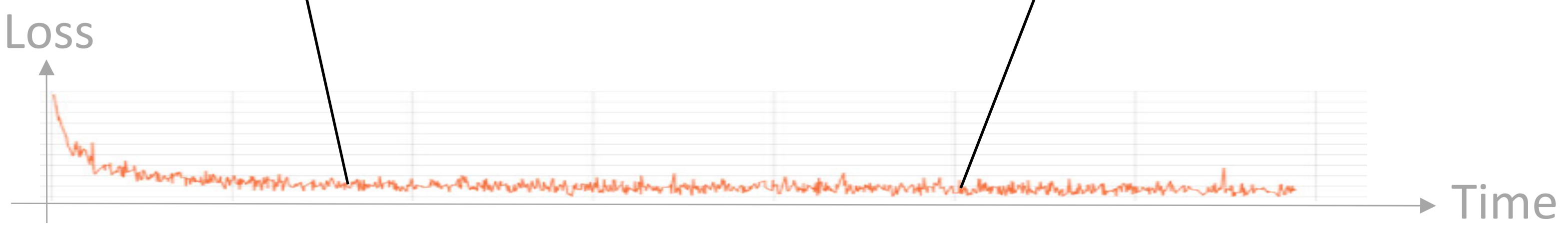
- Unsupervised Disentanglement
  - No guarantee in disentangling particular semantics

Different c

Rotation Angle

Different c

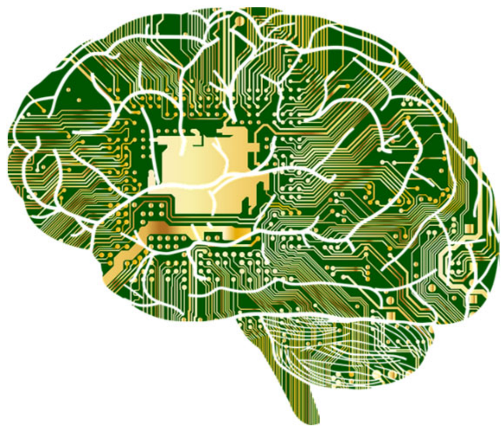
Width



Training process

Chen et al., InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets., NIPS 2016.

What We've Covered Today…



- Generative Models
  - Diffusion Model
- Transfer Learning
  - Visual Classification – Domain Adaptation
  - Visual Synthesis – Style Transfer
- Representation Disentanglement
  - Supervised vs. unsupervised feature disentanglement

Sprouts in the shape of text 'Imagen' coming out of a fairytale book.