01 _ Introduction

1. Network : $f = Wx$

2. Loss function : Softmax / Hinge a class
   optimal ⇒ 0 saturates / well enough

3. Activation function : Sigmoid / tanh / ReLU / Maxout
   saturated Stil trained Dead Relu zero centerd

4. Optimization : GD / SGD / GD with Momentum / RMSprop / Adam
   learning rate velocity combine the momentum
   mini batch Second Momentum

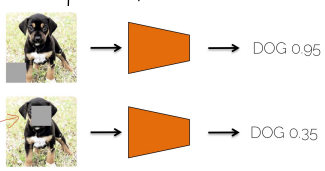
5. Training : Learning rate α / Over- and underfitting / split data

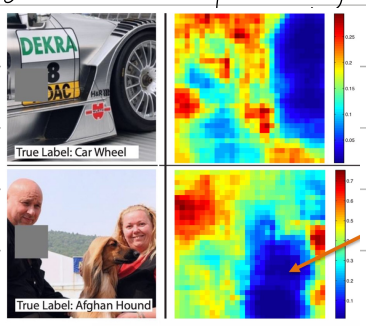
6. Regularization : Low Validation error / High training error
   Data augmentation / Early stopping / Bagging and Embedding / Dropout

7. Image task : Image filters / CNN / Pooling
   LeNet / Alex Net / VGG Net / Res Net / Inception layer / Google Net

8 Training trick : Setup Visualization / Overfit a sample → 0 loss, 100% correct
   Iteration and training time check
   Small network → debug → bigger Network / Sample
   ↳ change only one time

9 Visualization : ① Image space let
   pick a unit → Find maximize unit activation 9 image patches
   ② Occlusion expreiment
   Block different part in image → check network catch the importance
   ⇒ Map / point-wise / classification probability





③ t-SNE full image Representation

Motivation: { visualize the last FC layer
Nearest neighbor visualization based on L_2

⇒ Map high-dim embedding into 2D Map

02 – Siamese and Similarity Learning

1. ML ⇒ Classification, Regression, Similarity Learning { Comparison, Ranking } — comparison

Problem of Similarity Learning: Scalability: retrain model for different people — new

2. Similarity Function: $d(A,B) > t \Rightarrow$ not same / $d(A,B) < t \Rightarrow$ same

3. Siamese NN: Siamese Networ share weights ⇒ $f(A)$ and $f(B)$ ... $f(n)$ — Sai 'mee — encoding of image

4. Contrastive Loss: $d(A,B) = \|f(A) - f(B)\|^2 = L(A,B) \Leftrightarrow$ positive

comparison $\max(0, m^2 - \|f(A) - f(B)\|^2) = L(A,B) \Leftrightarrow$ negative

already far away → not pull them together

positive samples → $L(A,B) = y^* \|f(A) - f(B)\|^2 + (1 - y^*) \max(0, m^2 - \|f(A) - f(B)\|^2) \Leftrightarrow$ CE

to same points — positive pair — negative pair — 2 input — 1 input

$\|f(A) - f(P)\|^2 \leq \|f(A) - f(N)\|^2$

5. Triplet Loss: $L(A, P, N) = \max(0, \|f(A) - f(p)\|^2 - \|f(A) - f(n)\|^2 + m)$

Ranking

encourage margin between cluster → Hard negative mining: training with hard case

Train a few epoches ⇒ hard case $d(A,P) \approx d(A,N) \Rightarrow$ refine training

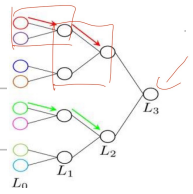
6. Improving Similarity learning — long time / local minima

- Loss:
  - Contrastive vs. triplet loss
- Sampling:
  - Choosing the best triplets to train with, sample the space wisely = diversity of classes + hard cases
- Ensembles:
  - Why not using several networks, each of them trained with a subset of triplets?
- Can we use a classification loss for similarity learning?

Sampling : Hierarchical Tree

leave ⇔ image class

recursively merge ⇒ root



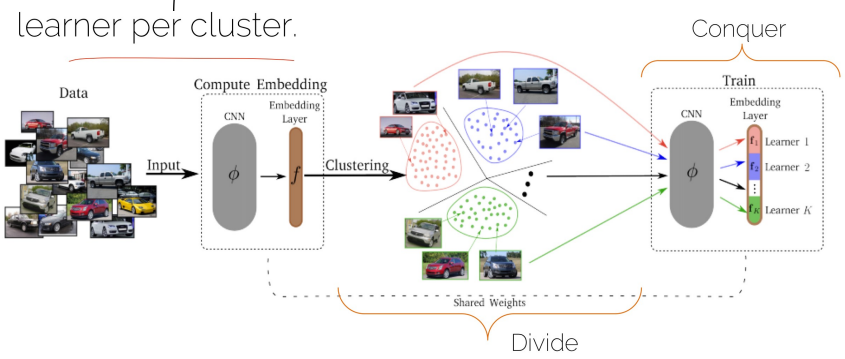
distance : $d(p,q) = \frac{1}{n_p n_q} \sum_{i \in p, j \in q} \|r_i - r_j\|$

① Randomly select l' nodes at 0 level
② select m-1 nearest classes based on d
③ t images per class are randomly collected

⇒ learn discriminative features from similar classes

$$\mathcal{L}_{\mathcal{M}} = \frac{1}{2Z_{\mathcal{M}}} \sum_{\mathcal{T}^z \in \mathcal{T}^{\mathcal{M}}} [\|\boldsymbol{x}_a^z - \boldsymbol{x}_p^z\| - \|\boldsymbol{x}_a^z - \boldsymbol{x}_n^z\| + \alpha_z]_+$$

Ensemble : devide spae into k cluster , each class set a learner



classification loss : For similarity learning

Tips and tricks

7 Application in Vision : Siamese on MNIST

Image correspondence : Objection recognition / ...

3D correspondence

Image retrieval

Self-supervised Learning : Learn from video

Automatically generate label

Optical flow

↳ FlowNet with CNN

↳ Siamese Network / Correlation layer

↓

Find image correspondence

03 _ AE and VAE

1. ML : Supervised Learning : Label / Find a mapping / Classification , Regression
   Unsupervised Learning : No label / Find data structure / Clustering

2. AE : Learning a lower-dimensional feature representation from unlabeled data
   Reconstruction Loss L_1, L_2
   Latent space $z \Rightarrow dim(z) < dim(x)$

3 AE for pre-training : Large unlabeled data / Small labeled data
   AE learn the features present
   Unsupervised train AE → Supervised train encoder with label

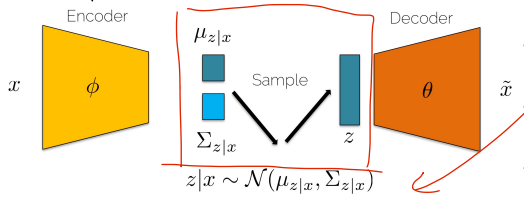
4. AE for pixel-wise prediction : FCN → SegNet (convolution + Layer / Upsampling + convolution)

   Transposed convolution : Unpooling + Convdution
   Upsampling : Interpolation / Interpolation + Convs / DeconvNet
   unpooling
   ↓
   Maxpooling

   UNet : Skip connections = pass the low level informations
   Convolution = pass the high level informations

5. AE in Vision : ① Semantic segmentation (SegNet)
   ② depth estimation (Stereo camera / Monocular depth)
   ③ Image Super Resolution (Low Resolution → high / Learning residual

6 VAE : Sample from the latent distribution to generate new output



Latent space is a Ganssian Distribution

Loss function :

$$= \boldsymbol{E}_z[\log p_\theta(x_i|z)] - \boldsymbol{E}_z\left[\log \frac{q_\phi(z|x_i)}{p_\theta(z)}\right] + \boldsymbol{E}_z\left[\log \frac{q_\phi(z|x_i)}{p_\theta(z|x_i)}\right]$$

$$= \boldsymbol{E}_z[\log p_\theta(x_i|z)] - KL(q_\phi(z|x_i)||p_\theta(z)) + KL(q_\phi(z|x_i)||p_\theta(z|x_i))$$

lower bound $= L(x_i, \phi, \theta)$ — $KL \geq 0$

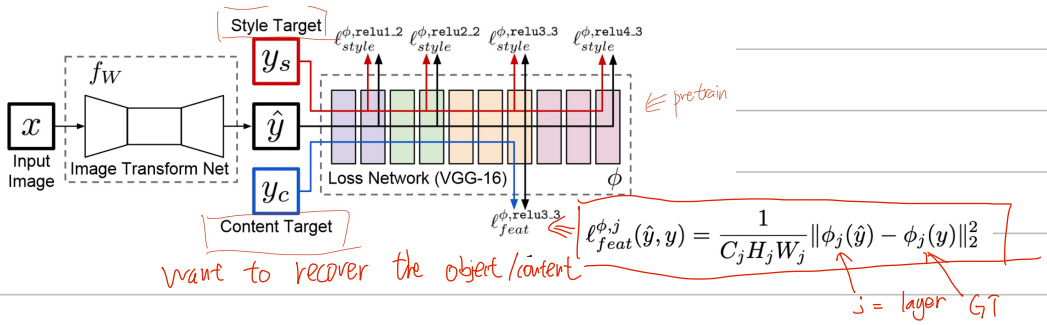
Optimize $\phi^*, \theta^* = \arg\max \sum_{i=1}^{N} L(x_i, \phi, \theta)$

7. Image synthesis : Semantic Segmentation image → Real image

perceptual loss : L_2 loss will penalize realistic result

Content loss measure the content of the image

Feature representation similarity



$$\ell^{\phi,j}_{feat}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$$

want to recover the object / content

j = layer GT

Style transfer: Content image + Style image = Style Transfer

$$\ell^{\phi,j}_{style}(\hat{y}, y) = \|G^{\phi}_j(\hat{y}) - G^{\phi}_j(y)\|_F^2$$

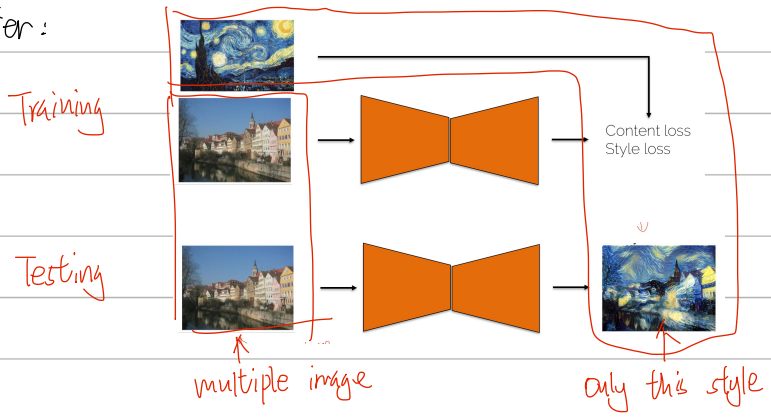
Gram matrix

$$G^{\phi}_j(x)_{c,c'} = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} \phi_j(x)_{h,w,c} \phi_j(x)_{h,w,c'}$$

Capture the information about which features tend to activate together

Loss perserve the stylistic features but not content

Fast transfer:



04_ Representation Learning

encode priors about data distributions

1. Representation Learning : Transform the raw data into a representation
   - Smoothness: close inputs map to close outputs
   - Compactness: input dimension >> output dimension
   - Robustness: features are insensitive to input noise
   - Abstraction and invariances -> problem driven

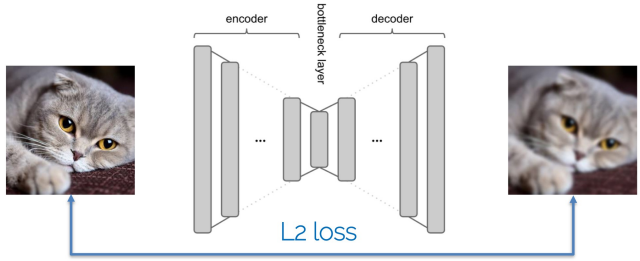
Convert the observation in the real world → mathematical form

↳ Feature vector ⇒ Classification / Reconstruction / Generation
↳ Handcrafted attribute / Binary / Embedding vector

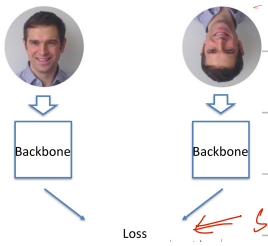
Supervised approach : ResNet + FC + Classifier
(FC: Feature vector; Classifier: cat)

Unsupervised approach : Clustering (k-means) ⇐ mean vector

Self-supervised approach : Data provide supervision
Loss force NN to learn features
Expensive to obtain annotations / Most data are unlabel

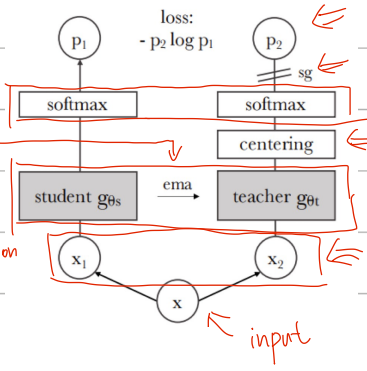


Self-supervised by Augmentation :



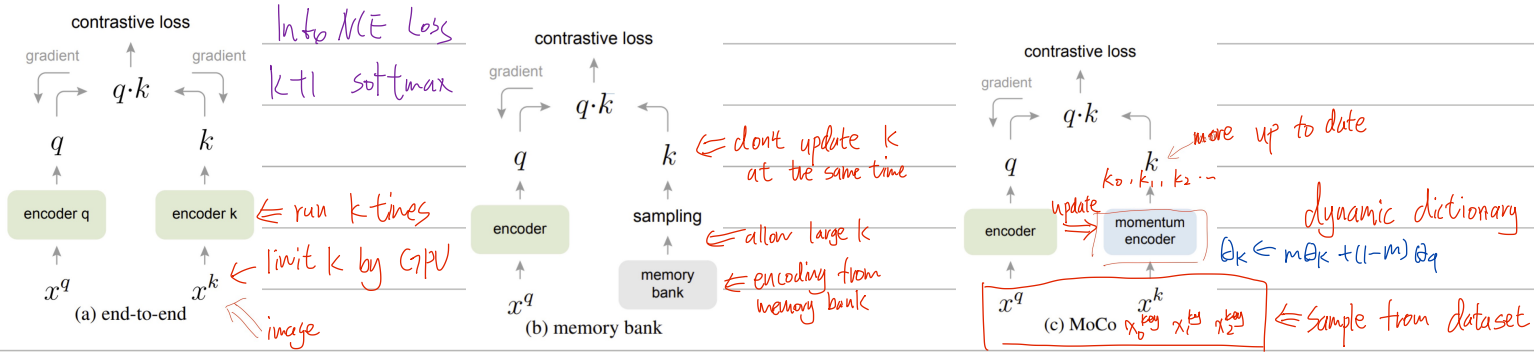
⇐ should be same

2. DINO: Self-distillation with no labels



⇐ Normalized features / CE loss
⇐ stop Gradient
temperature softmax ⇒
⇐ center over batch
Teacher params. are updated with an exponential moving average of student
Only update students ⇒
teachers is built from previous iterations of students
Students are for Feature extraction
⇐ same encoder, different params.
⇐ Augmentation
↖ input

3. Contrastive Learning : Learning embedding space, similiar samples pairs are close



Into NCE Loss

k+1 softmax

⇐ run k times

⇐ limit k by GPU

image

⇐ don't update k at the same time

⇐ allow large k

⇐ encoding from memory bank

more up to date

$k_0, k_1, k_2 \cdots$

dynamic dictionary

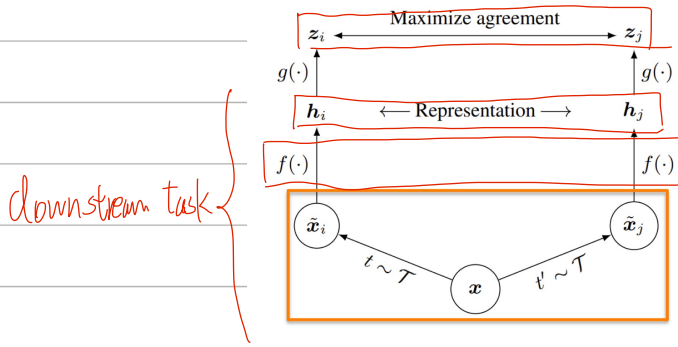
$\theta_k \leftarrow m\theta_k + (1-m)\theta_q$

$x_0^{key}\ x_1^{key}\ x_2^{key}$

⇐ Sample from dataset

4. Sim CLR : Simple Framework for Contrastive Learning of Visual Representation

Supervised via a contrastive loss in the latent space



⇐ Projected Representation

⇐ Visual Representation

⇐ Encoder

downstream task

⇐ argumented by two different operators

⇒ different Rotations

For a batch with N samples, 2N augmented samples are produced

Only one positive pair

2(N-1) negative pair

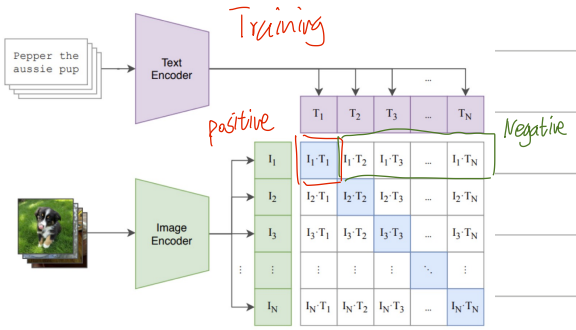
Large batch → negative pair

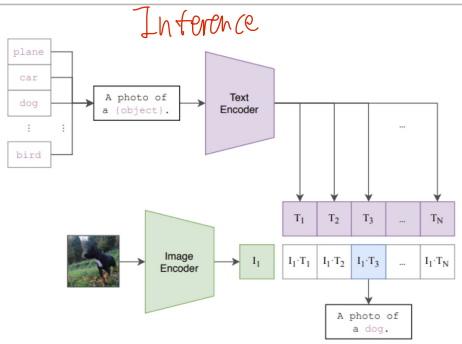
train : MLP on ResNet

test : ResNet

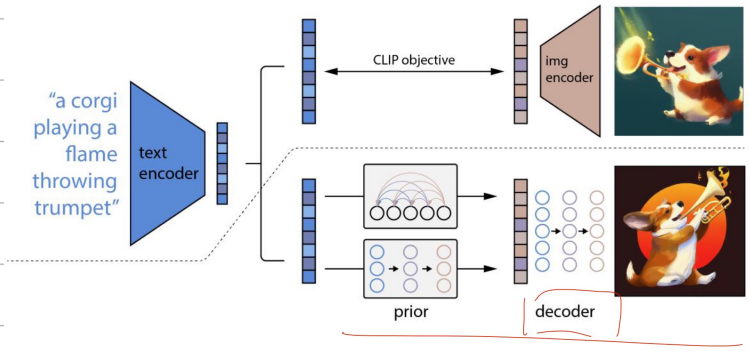
5. Multi-Model Representation Learning : image ⇔ text pair

CLIP: Contrastive Language-Image Pre-train





Application : text ⇒ image (DALL-E2)



Training

Inference

05 _ Sequence Models

1. Sequence Modelling : Text : Classification / Translation / Generation
   Image : Classification / Reconstruction
   Image patch

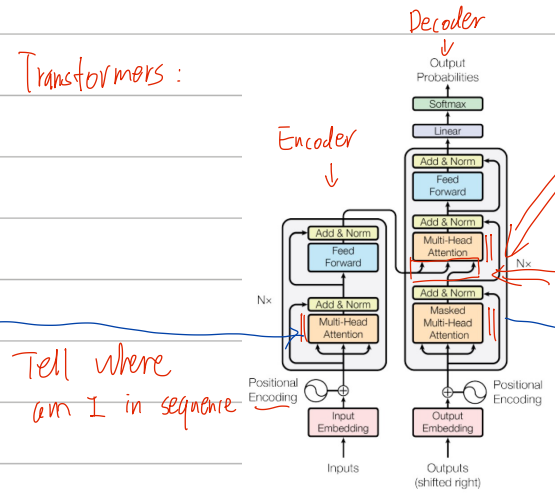
2 RNN : Recurrent Neural Network ⇒ short sequence / long-term dependency issue
LSTM : Long-short Term Memory ⇒ long-term sequence / still not for extreme long
All words are not equal important

3 Attention : Soft attention : Attend each part / sum(w) = 1 / Deterministic and differentiable
Hard attention : Attend one part / stochastic and non-differentiable / Monte-carlo → gradient
Self-attention : Attend input signal itself
Cross-attention : Attend other input signal as side information

4. Transformer in Language: Attention : $\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$

- Q: tell how important the Location is
- K: match the look-up table
- V: tell which word I should look up

Transformers :

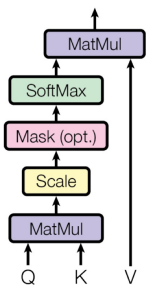


Not recurrent model
Multi-head Attention : Attend to different
input + output / cross attention
Tell where am I in sequence
Positional Encoding — int (1,2,3···)

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

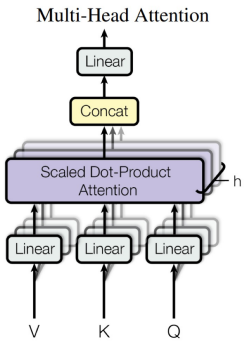
Scale dot-product attention



$$\text{Attention}(Q,k,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

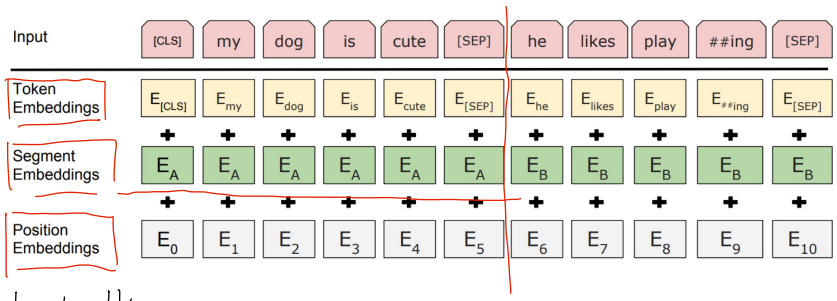
Triangular masking for unidirectional modelling : attend past
Full masking for bidirectional modelling : attend past and future

Multi-head Attention



Different heads attend to different parts

BERT : Big Transformer as text encoder



Pre-train 以

Two unsupervised task: { Masked Language Modelling (MLM)
Next Sentence Prediction (NSP)

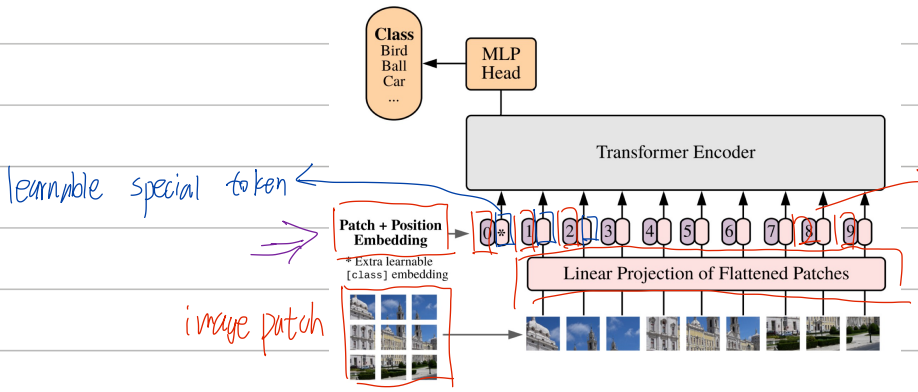
Masked Language Modelling (MLM)

↓ Randomly mask out some input word
↓ Predict the mask word

Next Sentence Prediction (NSP)

Take two sentence A and B
↓ 50% real B and 50% random B
↓ Predict B based on A

Transformer in CV : Vision Transformers (ViTs)



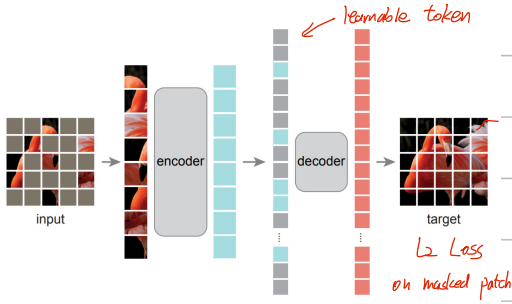
learnable special token

position of patch

MLP

image patch

Also can apply attention maps on input image



L2 Loss on masked patch

Masked AE : ViTs as AE

↓ Mask out some input image patch
↓ ViTs → Reconstructive image based on unmasked part

6. Transformer in Multimodal Learning : Vision - Language Transformers (ViLT)

Without supervision

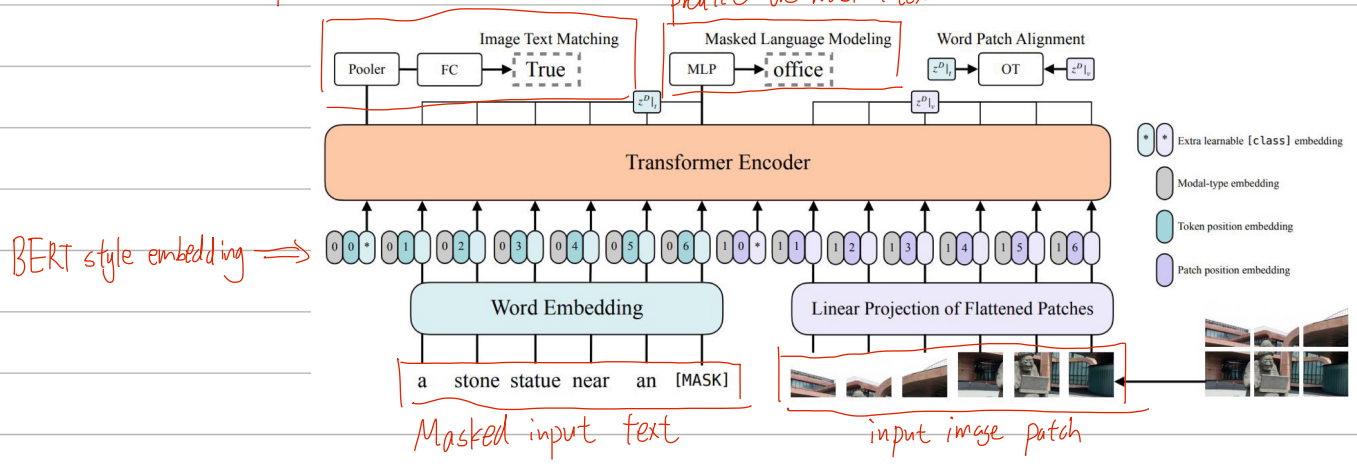
Input : concatenate image patches with text sequences

Per-taining with two self-supervised objectives

⇒ { Image text matching
Masked Language modelling

predict if image match text

predict the masked text



BERT style embedding ⟹

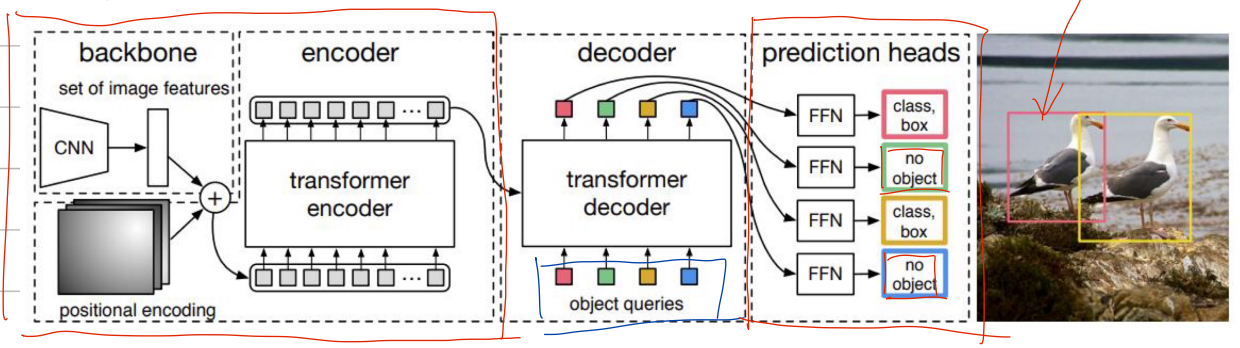
Masked input text

input image patch

7. Transformer in Object Detection : Detection Transformer (DETR)

on top of CNN

encode the input image into a sequence of image token



one-to-one match

decode object queries

predict object classes and bounding boxes

Matching via Hungarian Algorithm

Find prediction-GT assignments that minimize this cost

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} -\mathbb{1}_{\{c_i \neq \varnothing\}} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\sigma(i)})$$

L1 loss and generalized IoU loss

Predicted class probability ("no object" class excluded)

Hungarian Loss ⇒ (after bipartite matching)

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

L1 loss and generalized IoU loss

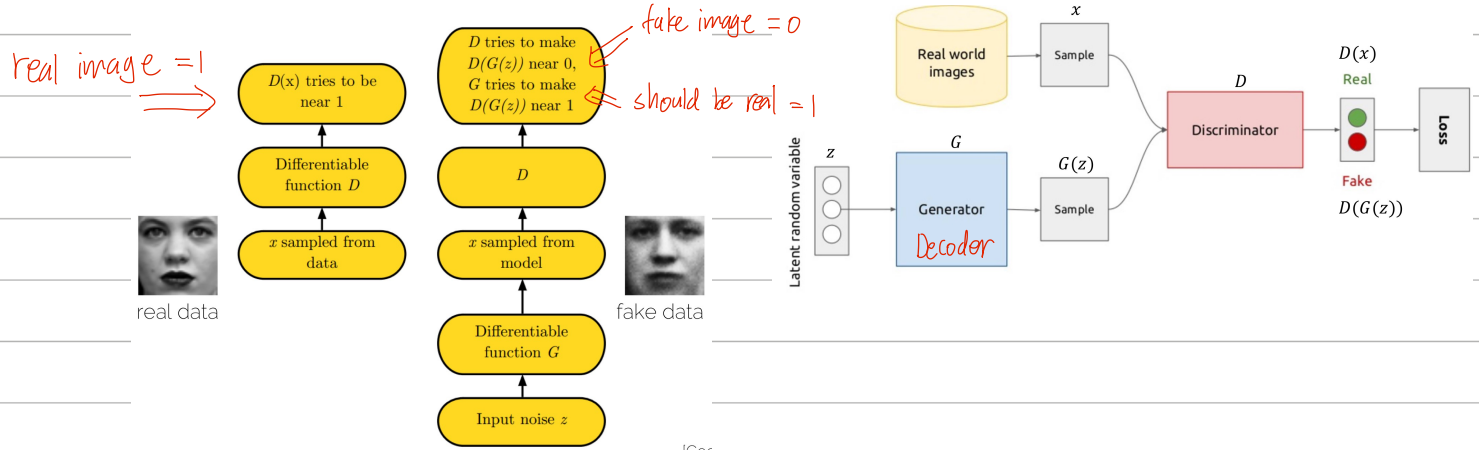
Predicted class probability (including the "no object" class)

06_GANs_01

1. Generative Models : Implicit density → Direct → GAN

2. Generative Adversarial Networks : Decoder in AE as Generative Model

Instead of using L2, want to learn a loss function



Loss Function

Discriminator loss

$$J^{(D)} = \underbrace{-\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)}_{\text{binary cross entropy}}$$

Generator loss

$$J^{(G)} = -J^{(D)}$$

G minimize probability that D is correct

Equilibrium is saddle point of discriminator loss

D supervise for G ≈ 0.5

Loss Function (Heuristic Method)

Fix G ↓ ↑ Fix D

Discriminator loss

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

Generator loss

$$J^{(G)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D\left(G(\boldsymbol{z})\right)$$

G maximize the log probability of D being mistaken

G can still learn even D reject all samples

Final Loss

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

* L(D) should never touch zero ⇒ no Gradient for G
  Loss curve should keep stable after some iterations
* Adaptive Schedule ⇒
  while loss_discriminator > t_d:
    train discriminator
  while loss_generator > t_g:
    train generator

* Balance needed ! weak D ⇒ No good gradient
  weak G ⇒ D always right

3 Mode Collapse : ⇒ no recover problem (restart training)

$$\min_G \max_D V(G,D) \neq \max_D \min_G V(G,D)$$

D in inner loop ⇒ convergence to correct dist

G in inner loop ⇒ convergence to one sample

More modes ⇒ small recovery rate e.g. GAN → Face

Large latent space ⇒ more mode collapse

Evaluation metrics : ① Human Evaluation : Visualization / train curves

② Inception Score (IS): Measure saliency and diversity

① Train a classifier

② Train a image generation model

③ Test the generation model by classifier

Saliency : check wheather generated images can be classified with high confindence

Diversity : check wheather obtain samples from all classes

③ Frechet Inception Distance (FID)

calculate the feature distance between the real and synthetic distribution

④ strong D ⇒ good G

...

4. GAN Hack : ① Normalized Input : between -1 and 1 / Tanh as the last layer

② Sampling : spherical z / Gaussian Distribution

③ Batch Norm :

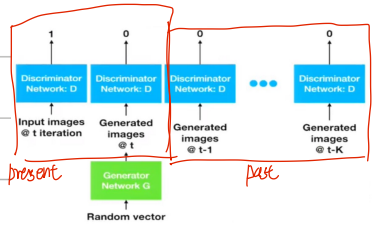
④ ADAM : SGD for D / ADAM for G

⑤ One-sided Label smoothing : prevent D give too large gradient to G

$$J^{(D)} = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}} \lambda \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log\left(1 - D\left(G(\boldsymbol{z})\right)\right)$$

Some value smaller than 1; e.g.,0.9

⑥ Historical Generator Batches : D also use historical generated images



⑦ Avoid Sparse Gradient : LeakyReLU / ...

⑧ D noisy : Due to SGD ⇒ exponential average of weights

4 Loss Function: Heuristic is standard / Loss alone will not affect the Network

① EBGAN: D is an AE

Reconstruction error of D(x) should low

Penalize D if reconstruction error below a value m

$$D(x) = ||Dec(Enc(x)) - x||$$

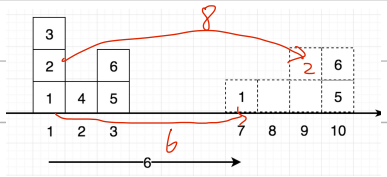
$$\mathcal{L}_D(x, z) = D(x) + [m - D(G(z))]^+$$

$$\mathcal{L}_G(z) = D(G(z))$$

$$\text{where } [\,u\,]^+ = max(0, u)$$

② BEGAN: Measure difference in data distribution of real and generated images

③ WGAN:



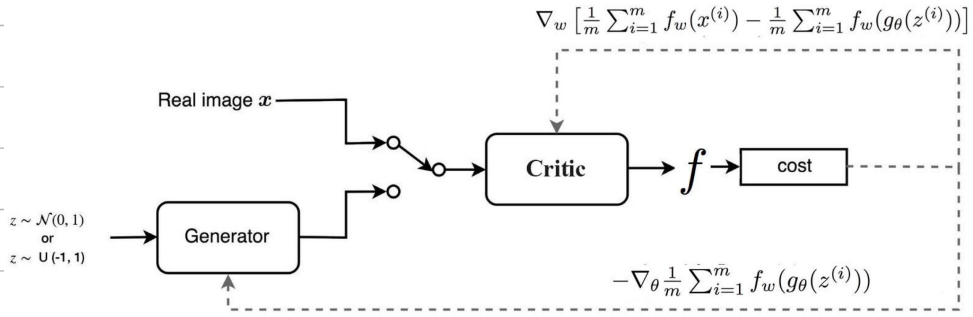
Two Expections should match

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)]$$

1-Lipschitz function: upper bound between densities

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2|.$$

Net work



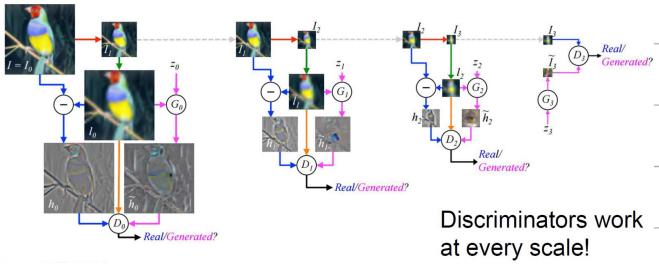
Comparison

	Discriminator/Critic	Generator
GAN	$\nabla_{\theta_d} \frac{1}{m}\sum_{i=1}^m \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right]$	$\nabla_{\theta_g} \frac{1}{m}\sum_{i=1}^m -\log\left(D\left(G\left(z^{(i)}\right)\right)\right)$
WGAN	$\nabla_w \frac{1}{m}\sum_{i=1}^m \left[f(x^{(i)}) - f(G\left(z^{(i)}\right))\right]$	$\nabla_\theta \frac{1}{m}\sum_{i=1}^m -f(G\left(z^{(i)}\right))$

Gratest improvement ⇒ WGAN Loss can convergence instead of stable

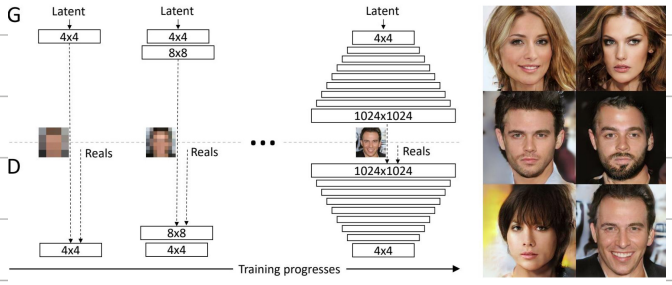
terrible weight cliping

5. GAN Architecture : ① Multiscale GANs



Discriminators work at every scale!

② Progressive Growing GANs



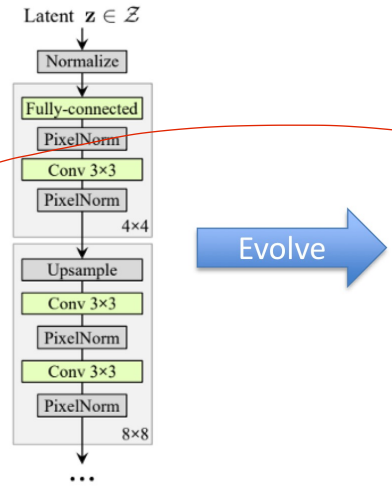
③ Style GAN

Modulation

Normalization

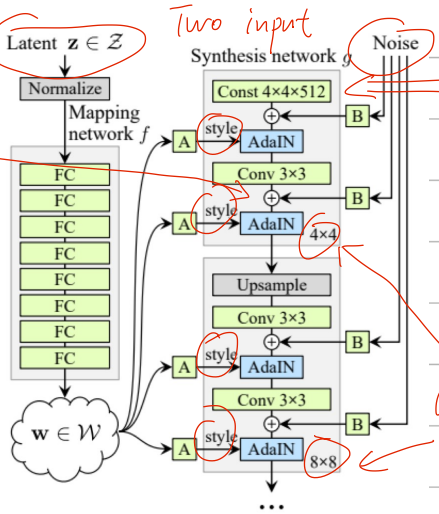
$$AdaIN(x,y) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \mu(y)$$

1. x: the activation from the previous layer
2. y: the style features (e.g. extracted from CNN) of your target style image
3. No trainable variables – mean and var directly calculated



Progressive GAN

Evolve



StyleGAN

Two input

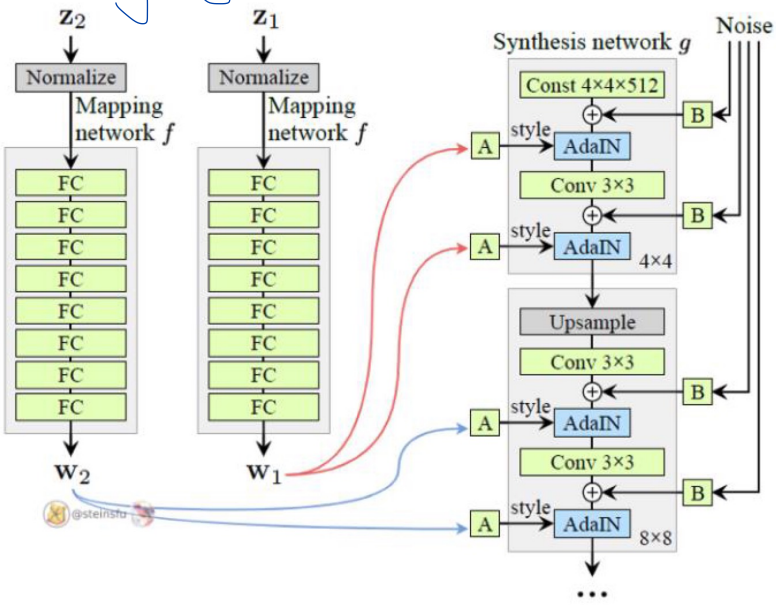
Constant input

different Resolution

StyleGAN2 better for limited data (No Augmentation)

StyleGAN3 No Aliasing

Mixing Regularization

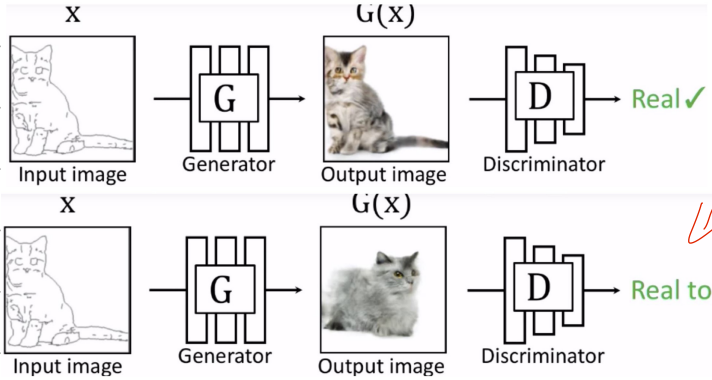


07 _ GANs _ 02

1. Conditional Generative Adversarial Networks (cGAN) : ① Gain control of output
② Modeling : add semantic meaning to latent space
③ Domain transfer : Label on A → transfer to B , train NN on B , test B

2 GAN Manifold : $G(z_0)$ → Linear interpolation in z space : $G(z_0 + t \cdot (z_1 - z_0))$ → $G(z_1)$

3 cGAN:



D can't see input label / condition
So give comparison based only on G and x

4 iGAN : ① Projection image on Manifold Input : real image x / output : latent vector z
↳ ① Optimization :

Optimization
$z^* = \arg\min \mathcal{L}(G(z), x^R)$
Reconstruction loss L | Generative model $G(z)$

② Inverse Network : Inverting Network $z = P(x)$ NN

$\theta_P^* = \arg\min_{\theta_P} \sum_{x_n^R} \mathcal{L}(G(P(x^R; \theta_P)), x^R)$

Auto-encoder
with a fixed decoder G

Prof. Niessner

③ Hybrid : Inverse Network + Optimization

② Manipulating the latent vector

constraint violation loss L_g | user guidance image

Objective: $z^* = \arg\min_{z \in \mathbb{Z}} \{ \sum_g (\mathcal{L}_g(G(z), v_g) + \lambda_s \cdot \|z - z_0\|_2^2 \}$.

data term | manifold smoothness

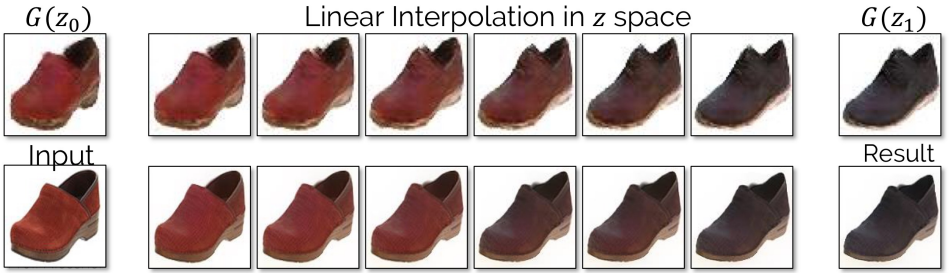
z_0 →

③ Edit Transfer

Motion (u, v)+ Color ($A_{3\times4}$): estimate per-pixel geometric and color variation

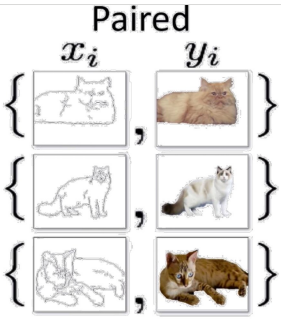
$$\iint \underbrace{\|I(x,y,t) - A\,I(x+u, y+v, t+1)\|^2}_{\text{data term}} + \underbrace{\sigma_s(\|\nabla u\|^2 + \|\nabla v\|^2)}_{\text{spatial reg}} + \underbrace{\sigma_c\|\nabla A\|^2}_{\text{color reg}} dxdy$$

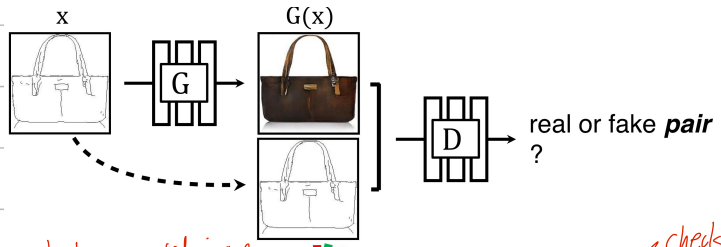
constraint motion　constraint colour



cGAN : Interactive GAN (User edit ⇒ generated (customerized image)

5 Mapping problem : ① Pair : Pix 2 Pix





check generated image with style/image

check input style/image with real image

$$\min_G \max_D \mathbb{E}_{x,y}[\log D(x, G(x)) + \log(1 - D(x, y))]$$

fake pair　real pair

match joint distribution p(G(x), y) ~ p(x, y)

Prof. Niessner　31

⇒ $L = L_{GAN} + \lambda L_1$　more constraint

Unet presene the structure

cGAN tend to ignore random vector Z

- L_1/L_2 for low frequency details
- D for high frequency details

Pix2PixHD

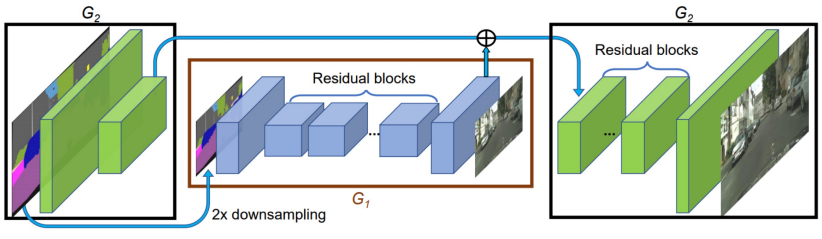
Expand Pix2Pix to multi-scale

same G and D but on different resolutions → large receptive fields

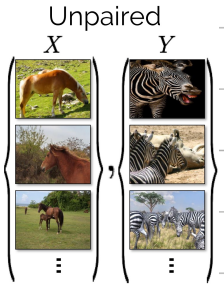
Use multi-scale D

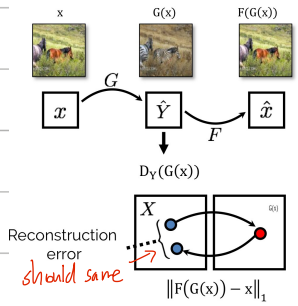
⇒ different receptive fields

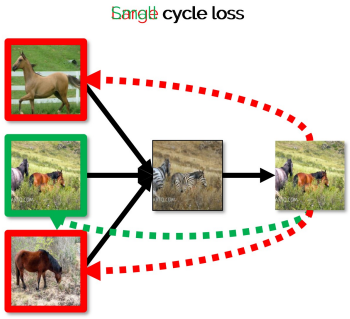
$$\min_G \max_{D_1,D_2,D_3} \sum_{k=1,2,3} L_{GAN}(G, D_k)$$

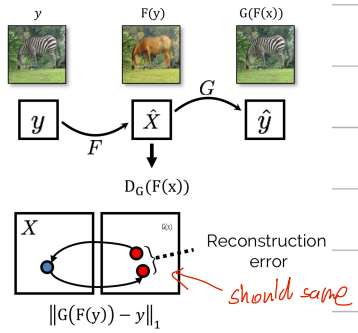


② Unpaired : Cycle-Consistent Adversial Network









GAN don't force output is as same as input
⇒ use F map ontput return to input

6 cGAN Application : Image Manipulation / Image Generation / Image Restoration / Image Interpolation
Image Understanding / Multi-model Manipulation / 3D Aware Synthesis