GAN in CV

By Wang Zirui
Deep Generative Model Study Group
Instructed by Prof. Tang of UMontreal
PKU, 2017.08.

Outlines

- Introduction
- Papers
- Related works for further reading

Introduction

Introduction

- Image Generation Tasks Definition
- Two Methods for Image Generation Tasks
- Different Losses
- Generator Architectures
- Result Evaluation

Tasks Definition

- Generate images that meet the task requirements, often with the given inputs

- ill-posed multi-modal problem
- probabilistic one-to-many mapping

Two Methods for Image Generation Tasks

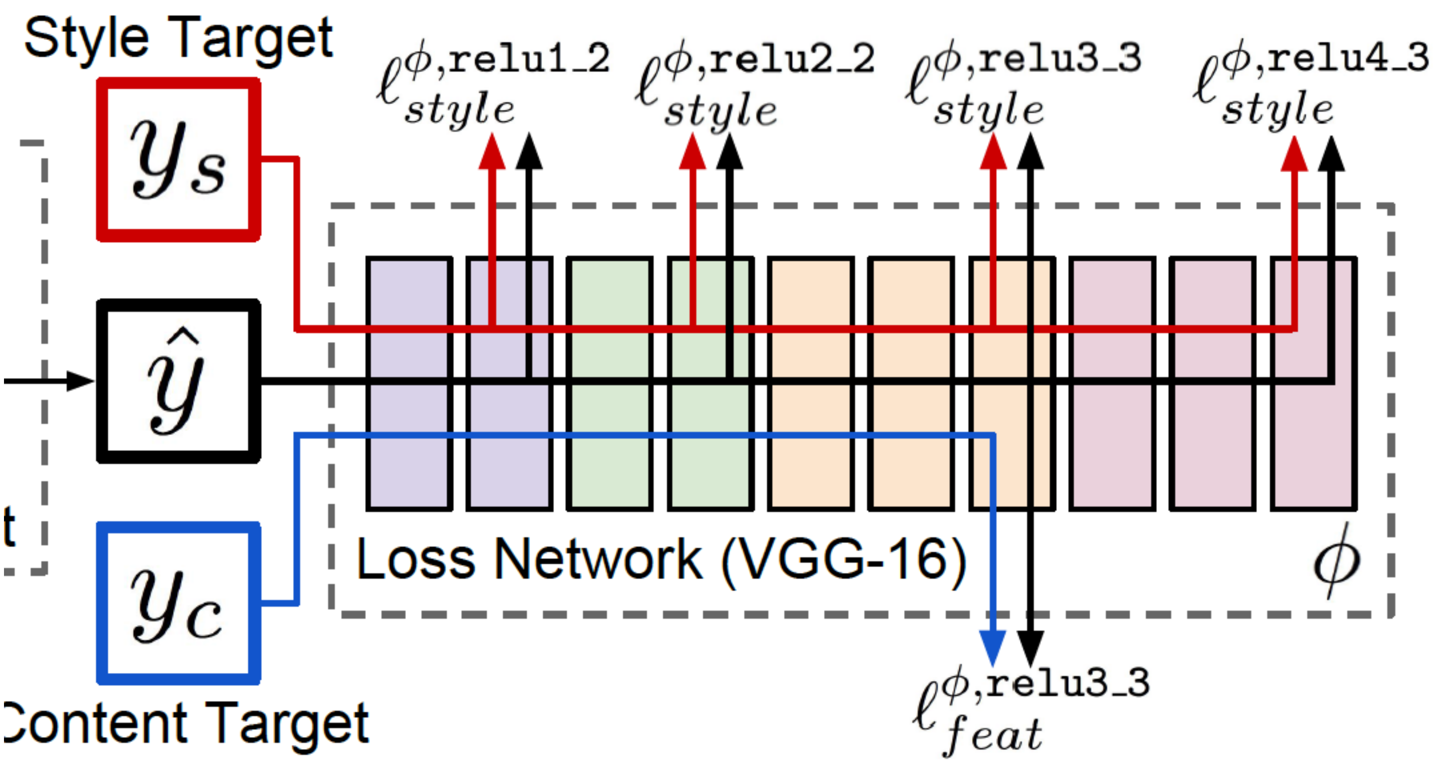
- Optimization-based

- Feed-forward network/Generator-based

Optimization-based

- Use DNN features to define losses.
- Use gradient descent to get optimal image.

- easy, flexible
- slow inference



Johnson et al. Perceptual Losses for Real-Time Style Transfer and Super-Resolution

Generator-based/Generative model

- Train a generator using specific losses

- need large training data
- most are fast in the inference time(exc. PixelCNN)
- adversarial training can be used

$$x_1 \sim \mathcal{X}_1 \longrightarrow \boxed{\mathbf{F}} \longrightarrow x = F(x_1)$$

Different Losses

- L2(mean square error)/L1 loss in image space
- Perceptual loss/VGG loss/Alex loss
- General adversarial loss
- Conditional adversarial loss

L2(mean square error)/L1 loss in Image Space

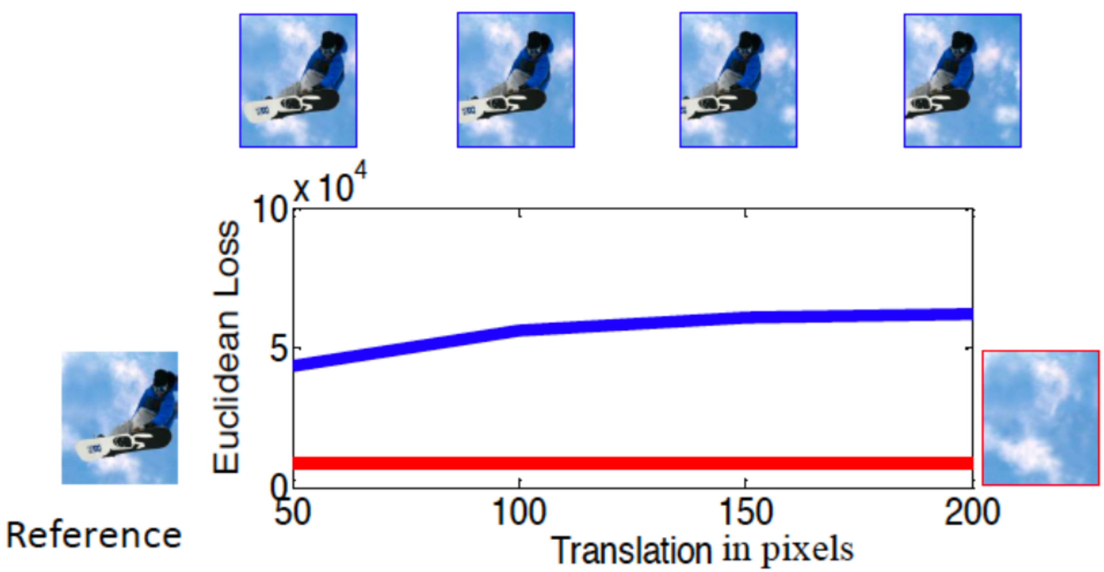
$$\mathcal{L}_p(X, Y) = \ell_p(G(X), Y) = \|G(X) - Y\|_p^p,$$

- low noise, smooth, but blurry
- changes like translation is not well expressed
- make average/median over possible answers

- L1 loss a little bit less blurry

Pitfall of Euclidean distance for image modeling

- Blue curve plots the Euclidean distance between a reference image and its horizontal translation.
- Red curve is the Euclidean distance between  and 



- Indeed, using the L2 loss comes from the assumption that the data is drawn from a Gaussian distribution, and works poorly with multimodal distributions. (Mathieu et al. 2016)

- Per-pixel regression treats the output space as "unstructured" in the sense that each output pixel is considered conditionally independent from all others given the input image(Isola et al. 2016)

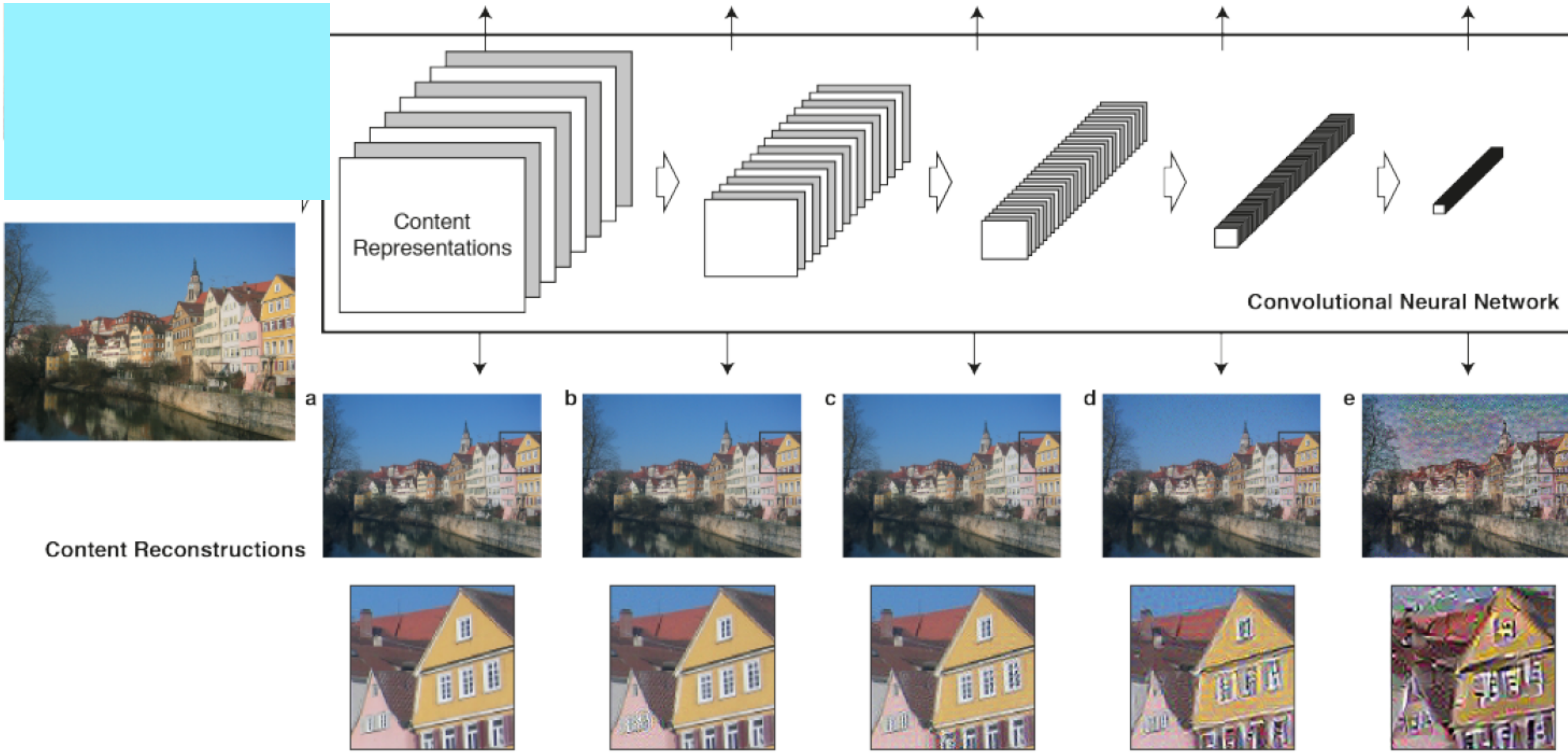
Perceptual loss(VGG loss/Alex loss)

$$\mathcal{L}_{feat} = \sum_i ||C(G_\theta(\mathbf{x}_i)) - C(\mathbf{y}_i)||_2^2.$$

- Capture high level features using a pre-trained model like VGG or AlexNet, then measure the distance in feature space
- Convolutional networks provide a feature representation with desirable properties. They are invariant to small smooth deformations, but sensitive to perceptually important image properties, for example sharp edges and textures(Dosovitskiy et al. 2016)

- Lose fine details, produce artifacts not natural or photo-realistic

- Since feature representations are typically contractive, many images, including non-natural ones, get mapped to the same feature vector(Dosovitskiy et al. 2016)



Gatys et al. 2015b

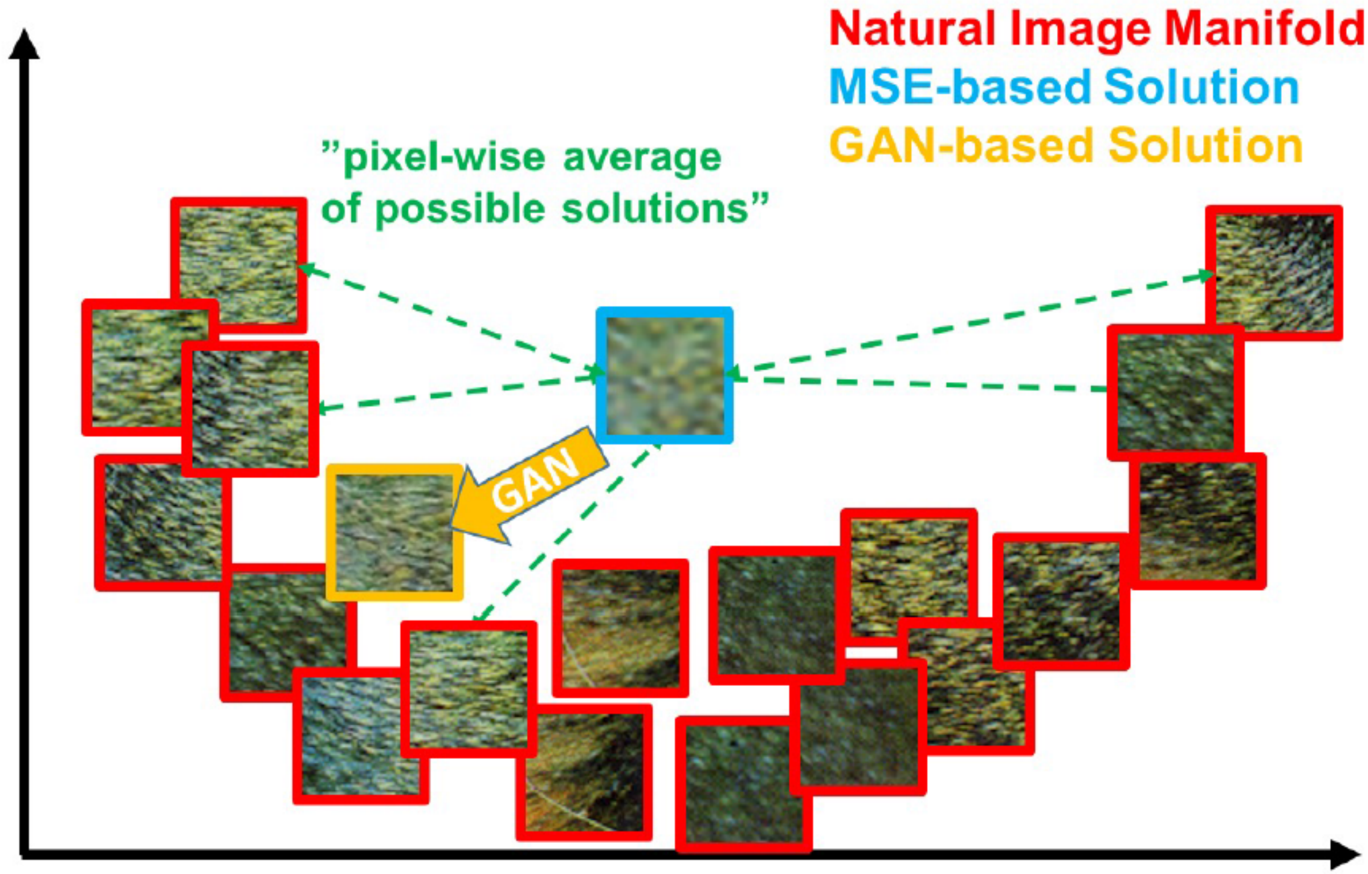
Why are GANs useful for computer vision?

Hand-crafted features → Deep Networks

Hand-crafted objective function → Generative Adversarial Networks

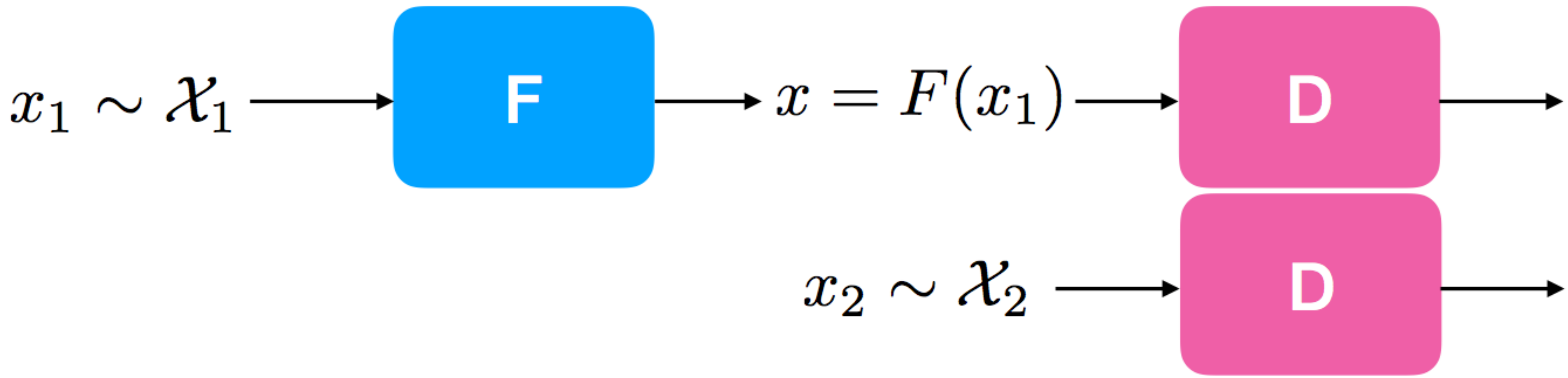
- Why are generated samples blurry? Difficult to hand-craft a good perceptual loss function

- Adversarial loss eliminates the need of hand-crafting objective functions for various computer vision problem.
- Forces the generated images to be indistinguishable from real images. This is “exactly” the objective that tasks aim to optimize.



(Ledig et al. CVPR'17)

General adversarial loss



$$\max_F E_{x_1 \sim p_{\mathcal{X}_1}}[\log D(F(x_1))]$$

$$\max_D E_{x_2 \sim p_{\mathcal{X}_2}}[\log D(x_2)] + E_{x_1 \sim p_{\mathcal{X}_1}}[\log(1 - D(F(x_1)))]$$

Ming-Yu Liu et al. CVPR 2017 GAN Tutorial

- Constrain the output images on the natural(answer) image manifold

- Make results sharp and realistic
- Denoise and get rid of artifacts

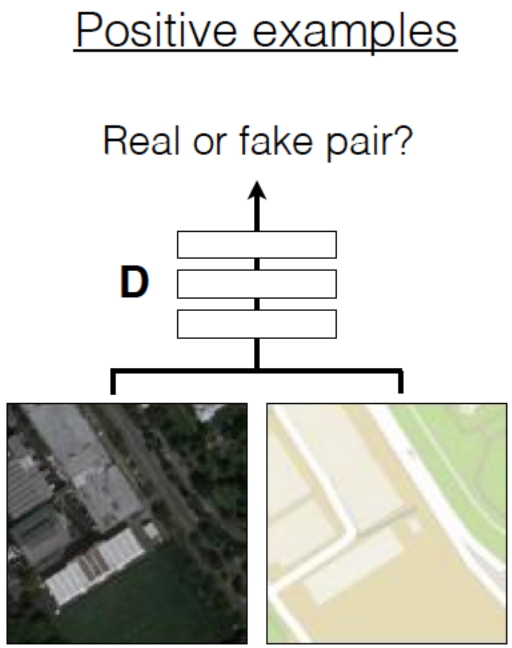
- Need other losses to explicitly constrain the input-output relationship

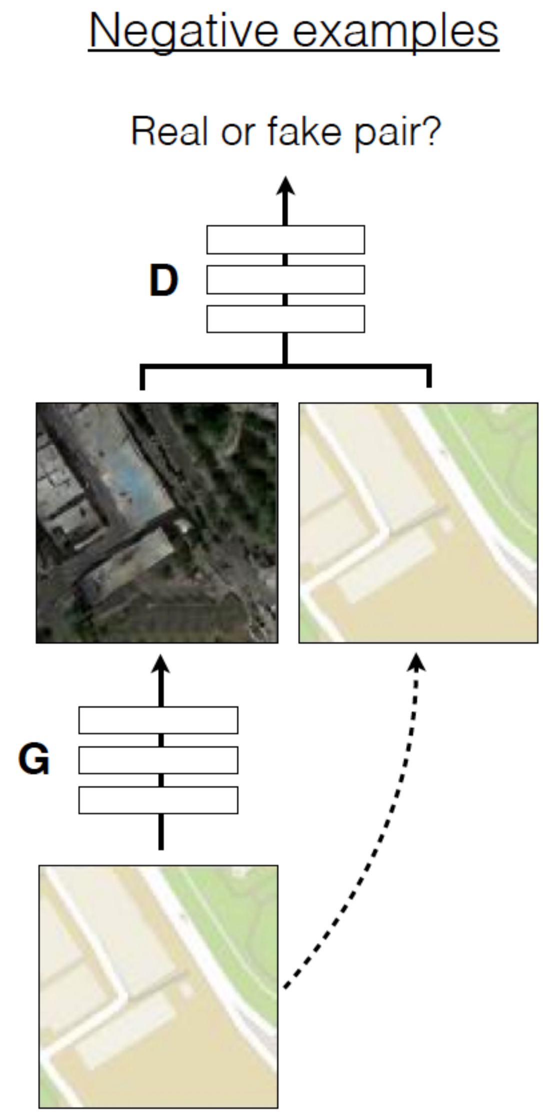
- Adversarial loss is difficult to train and unstable. MSE loss proved to be useful as it stabilizes and accelerates training

Conditional Adversarial Loss

- Used for supervised image generation. Give both input and output to the discriminator.

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y\sim p_{data}(x,y)}[\log D(x, y)] + \mathbb{E}_{x\sim p_{data}(x), z\sim p_z(z)}[\log(1 - D(x, G(x, z))]$$





G tries to synthesize fake images that fool **D**

D tries to identify the fakes

Isola et al. 2016

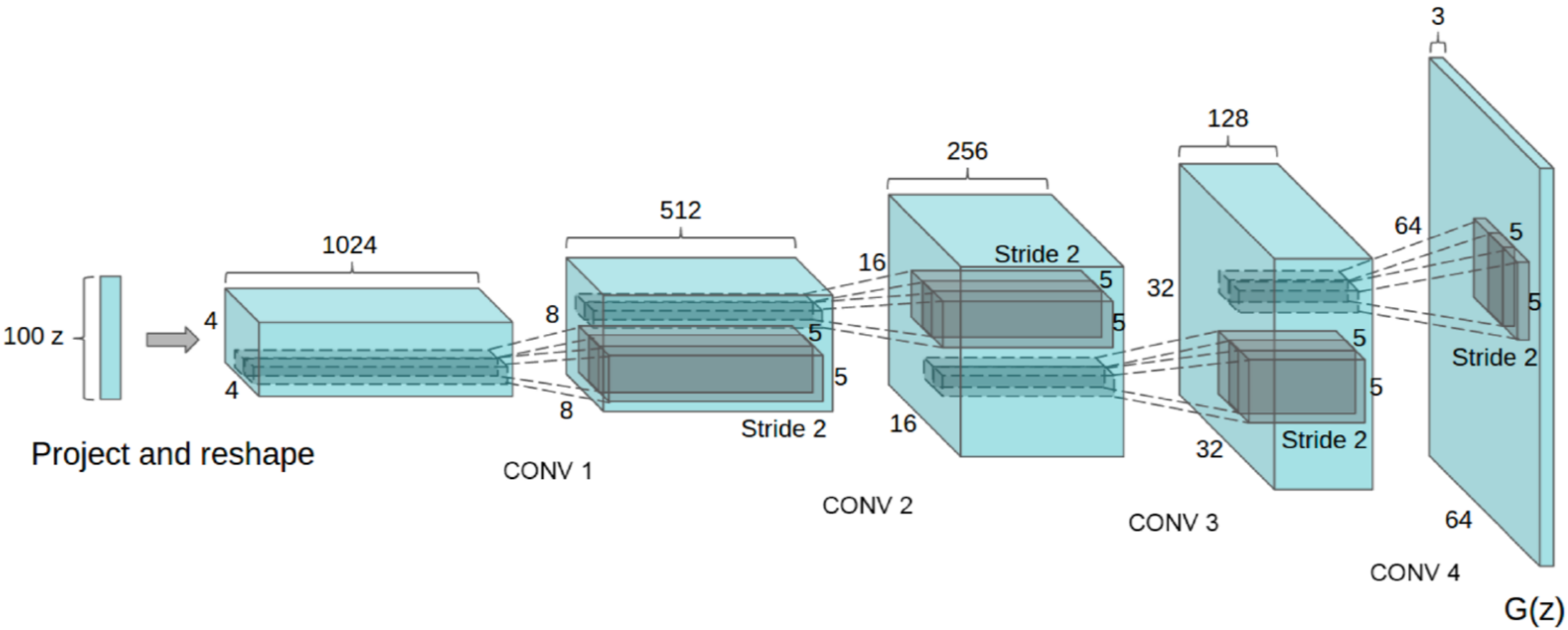
- Not only make outputs sharp and natural, but as close to correct answers as possible corresponding to the input
- Actually model a joint distribution

- MSE also helps training

Generator Architecture

- (conv +) deconv (+ skip)
- multi-scale

DCGAN



Radford et al. 2016

- (encoder+)decoder
- Skip connection to preserve the high frequency information
- Kind of like attention

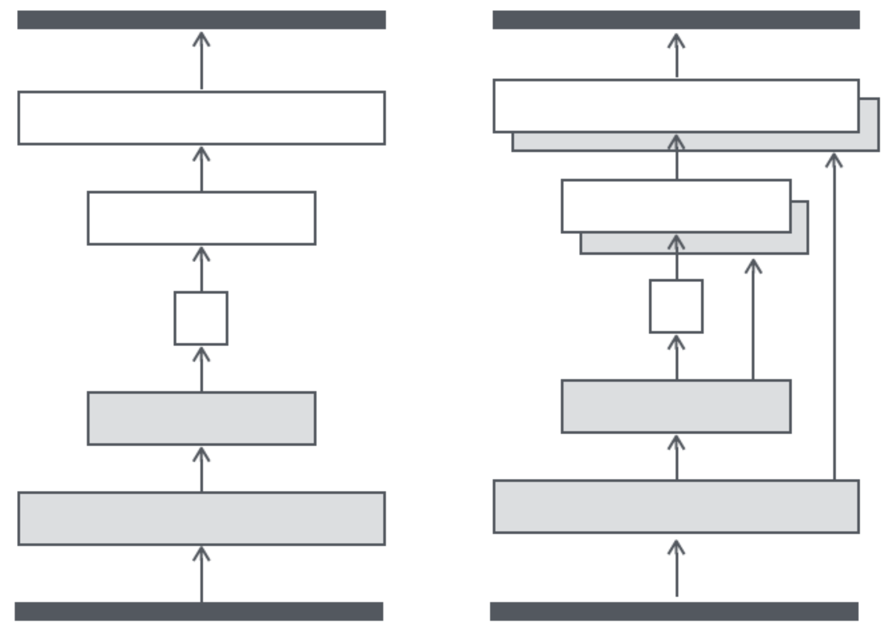
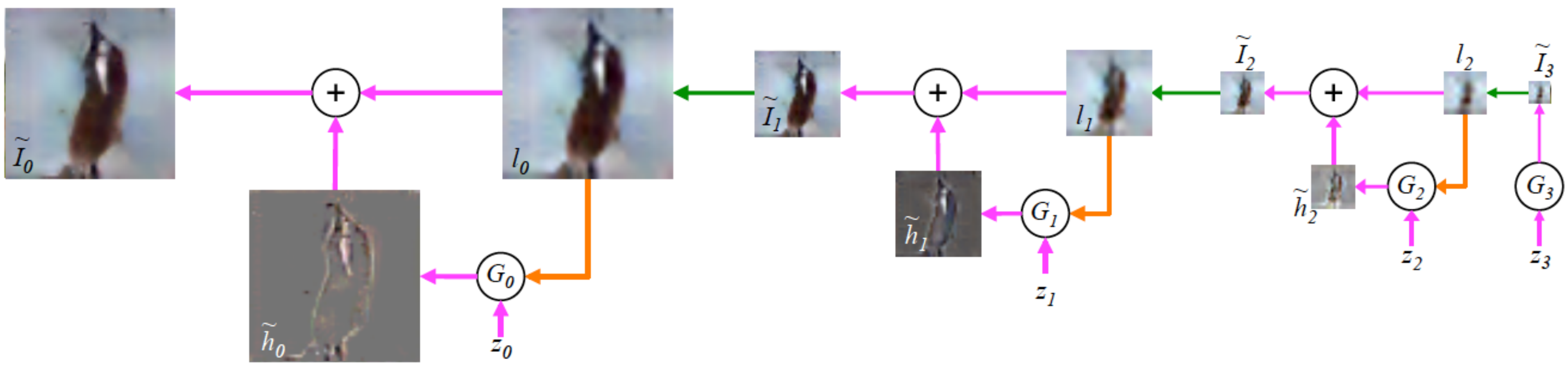


Image credit, Isola et al. 2016

LAPGAN: Multi-scale



Denton et al. 2015

Result Evaluation

- No good quality metrics now

- PSNR/SSIM prefer MSE
- Human evaluation: Amazon Mechanical Turk
- FCN score: use pre-trained semantic segmentation classifier to evaluate the similarity of ground truth and generated image

Papers

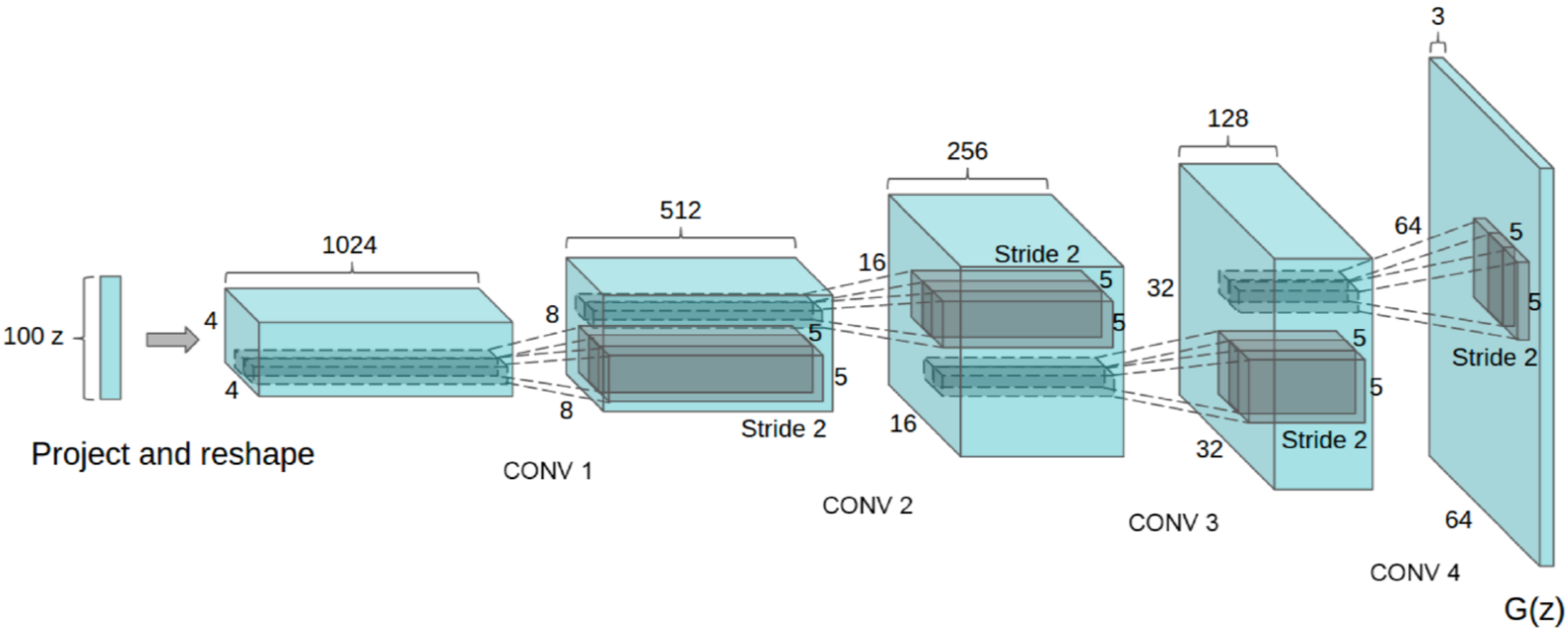
Unsupervised representation learning with deep convolutional generative adversarial networks

DCGAN

Radford et al.
ICLR, 2016

arXiv:1511.06434

- Propose a class of architectures that make training process stable



Architecture guidelines for stable Deep Convolutional GANs

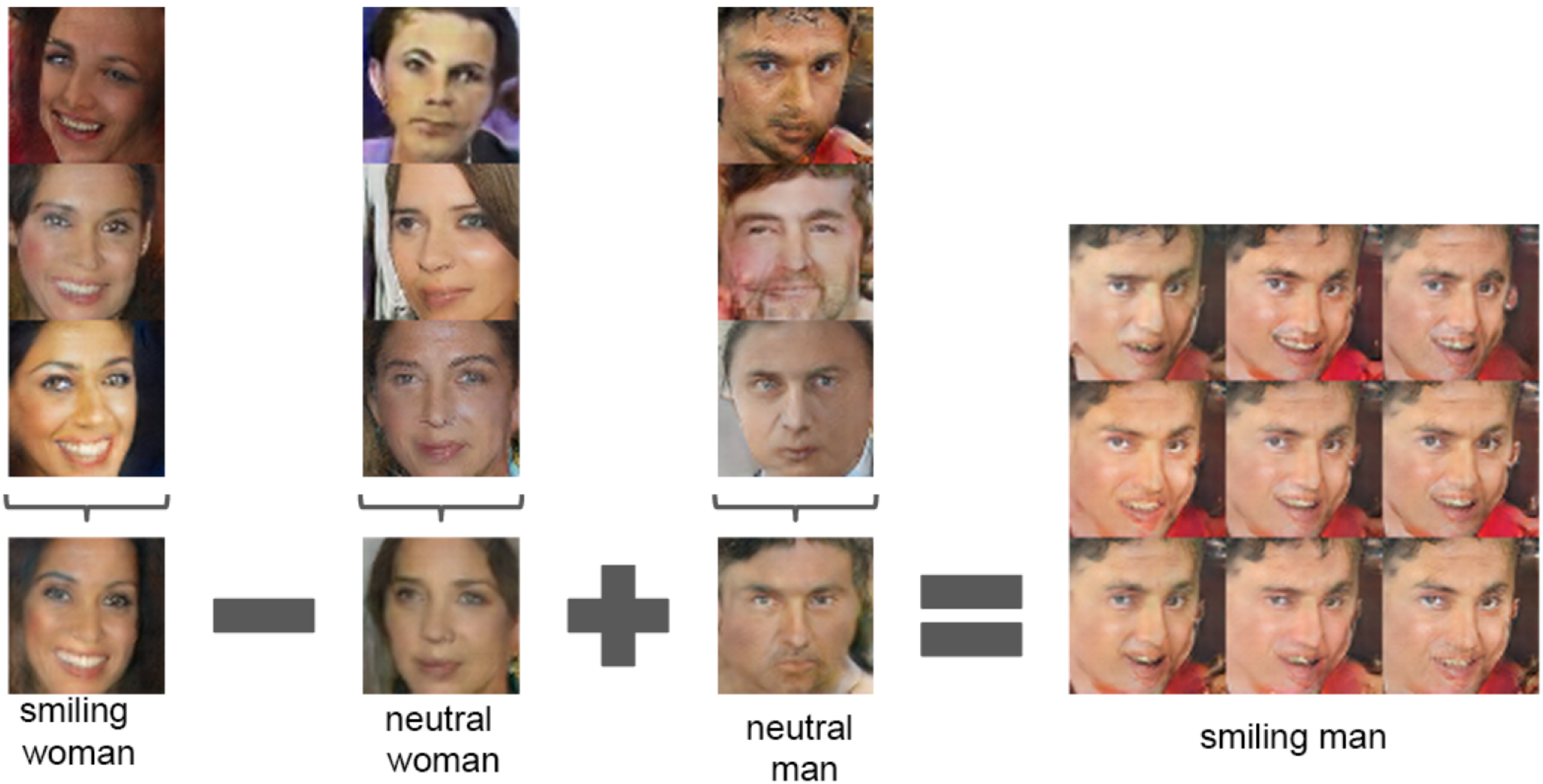
- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

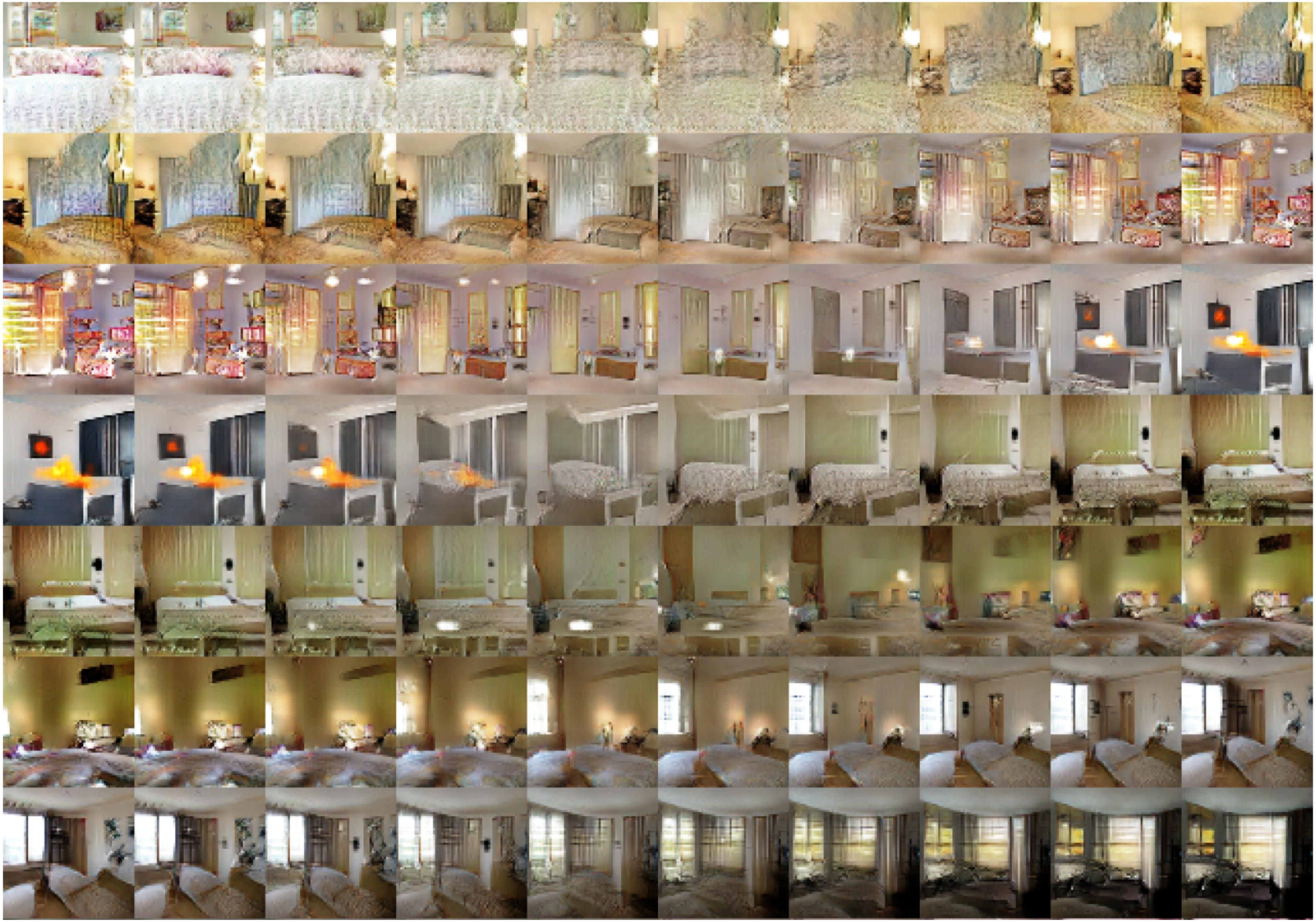
- Use trained discriminator as unsupervised feature extractor

Table 1: CIFAR-10 classification results using our pre-trained model. Our DCGAN is not pre-trained on CIFAR-10, but on Imagenet-1k, and the features are used to classify CIFAR-10 images.

Model	Accuracy	Accuracy (400 per class)	max # of features units
1 Layer K-means	80.6%	63.7% (±0.7%)	4800
3 Layer K-means Learned RF	82.0%	70.7% (±0.7%)	3200
View Invariant K-means	81.9%	72.6% (±0.7%)	6400
Exemplar CNN	84.3%	77.4% (±0.2%)	1024
DCGAN (ours) + L2-SVM	82.8%	73.8% (±0.4%)	512

- Investigate the learned latent space





The smooth transition shows that the model has learned an interesting representation, not just memorizes examples.

Image Translation

Generating Images with Perceptual Similarity Metrics based on Deep Networks

DeePSIM

Dosovitskiy et al.

NIPS, 2016

arXiv:1602.02644

- Sum up 3 kinds of losses when given a supervised learning task and a training set of input-target pairs {xi, yi}

$$\mathcal{L} = \lambda_{feat}\,\mathcal{L}_{feat} + \lambda_{adv}\,\mathcal{L}_{adv} + \lambda_{img}\,\mathcal{L}_{img}.$$

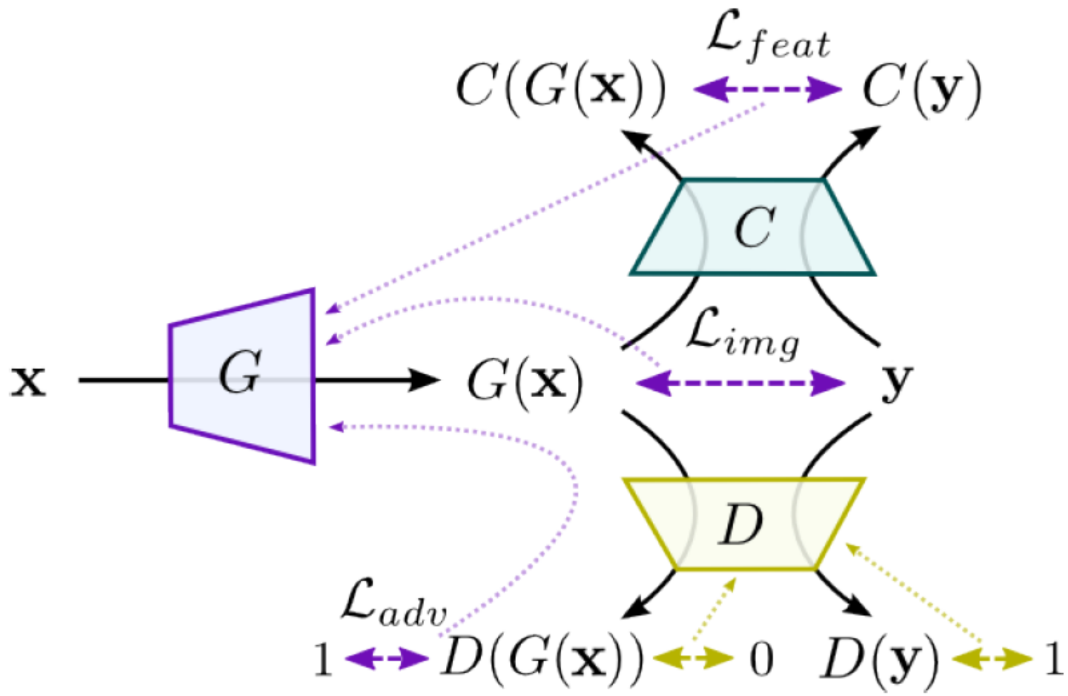


Figure 2: Schematic of our model. Black solid lines denote the forward pass. Dashed lines with arrows on both ends are the losses. Thin dashed lines denote the flow of gradients.

$$\mathcal{L}_{feat} = \sum_i ||C(G_\theta(\mathbf{x}_i)) - C(\mathbf{y}_i)||_2^2.$$

- “Since feature representations are typically contractive, many images, including non-natural ones, get mapped to the same feature vector. Hence, we must introduce a natural image prior.”

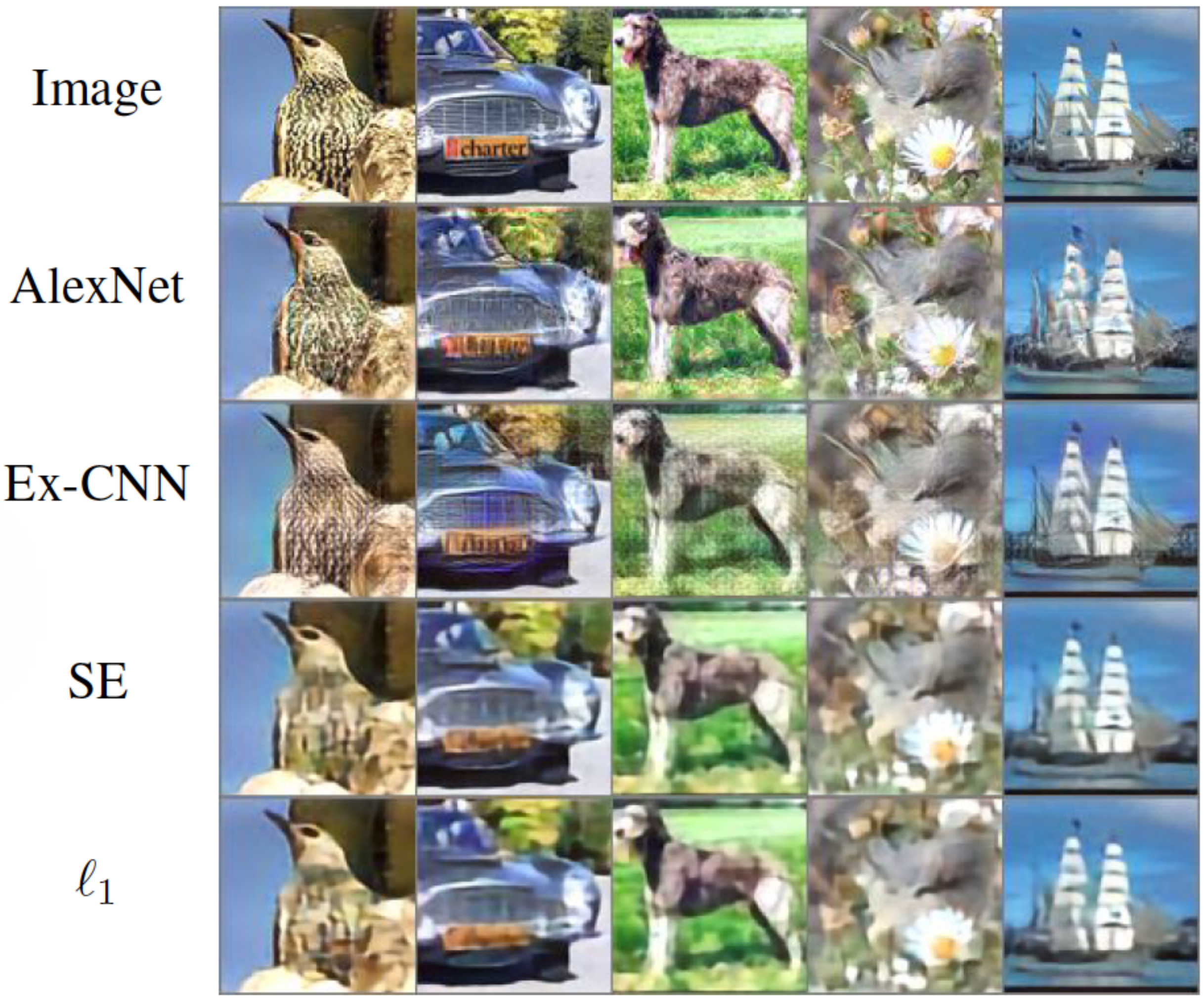
$$\mathcal{L}_{discr} = -\sum_i \log(D_\varphi(\mathbf{y}_i)) + \log(1 - D_\varphi(G_\theta(\mathbf{x}_i))),$$

$$\mathcal{L}_{adv} = -\sum_i \log D_\varphi(G_\theta(\mathbf{x}_i)).$$

- Adding a loss in the image space stabilize adversarial training

$$\mathcal{L}_{img} = \sum_i ||G_\theta(\mathbf{x}_i) - \mathbf{y}_i||_2^2.$$

Experiment1: Autoencoder with DeePSIM Loss



- Actually SE and L1 loss have lower Euclidean reconstruction error, which shows that Euclidean error doesn't mean better result quality.

SE loss	ℓ_1 loss	Our-ExCNN	Our-AlexNet
34.6 ± 0.6	35.7 ± 0.4	50.1 ± 0.5	52.3 ± 0.6

Table 4: Classification accuracy (in %) on STL with autoencoder features learned with different loss functions.

Experiment2: VAE with DeePSIM Loss

$$\sum_i -\mathbb{E}_{q(z|x_i)} \log p(x_i|z) + D_{KL}(q(z|x_i)||p(z)),$$

- If we assume that the decoder predicts a Gaussian distribution at each pixel, then it(log likelihood) reduces to squared Euclidean error in the image space.
- Replace the first term with DeePSIM
- Just like VAE-GAN

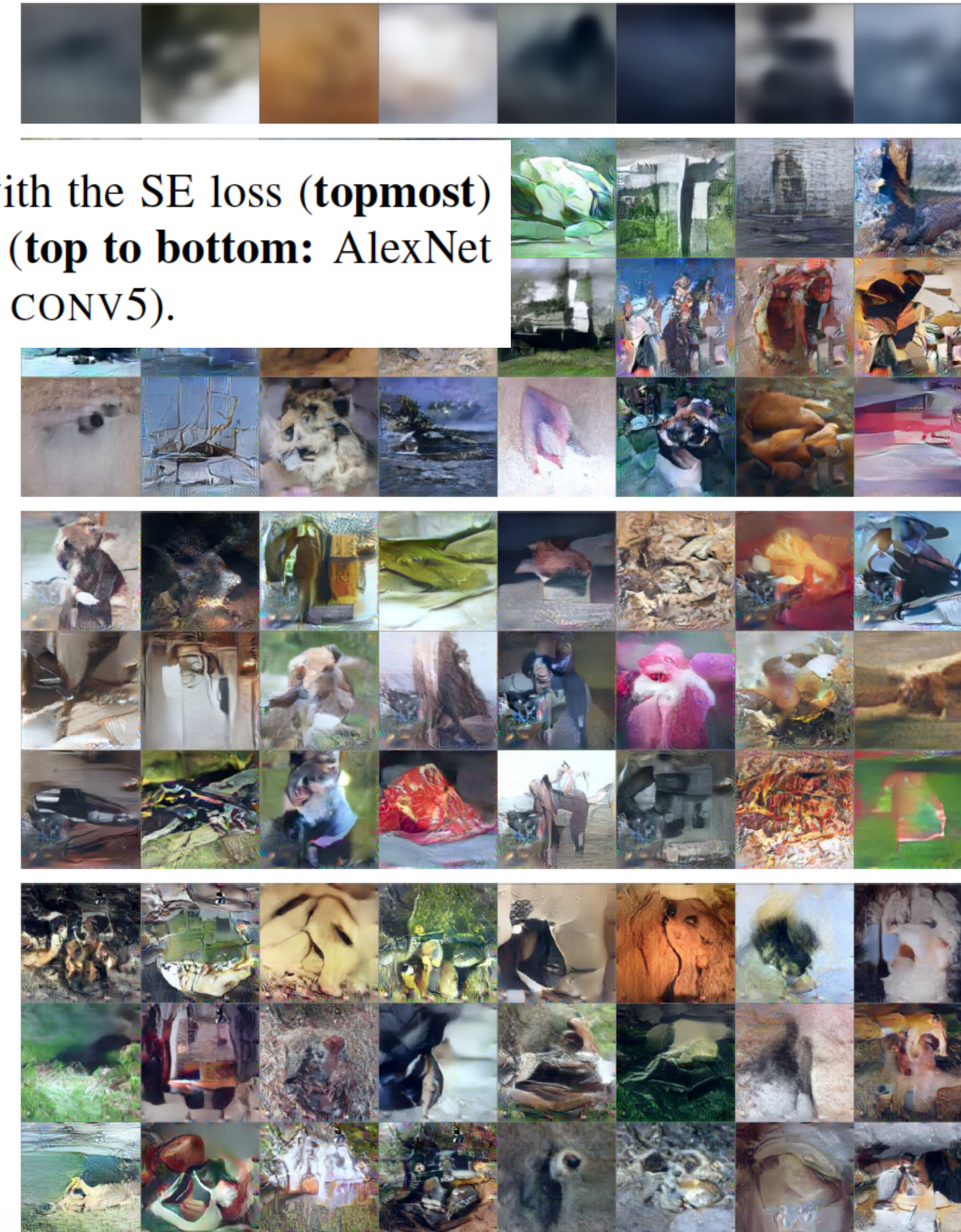


Figure 4: Samples from VAE with the SE loss (**topmost**) and the proposed DeePSiM loss (**top to bottom:** AlexNet CONV5, AlexNet FC6, VideoNet CONV5).

Experiment3: Invert AlexNet with DeePSIM Loss

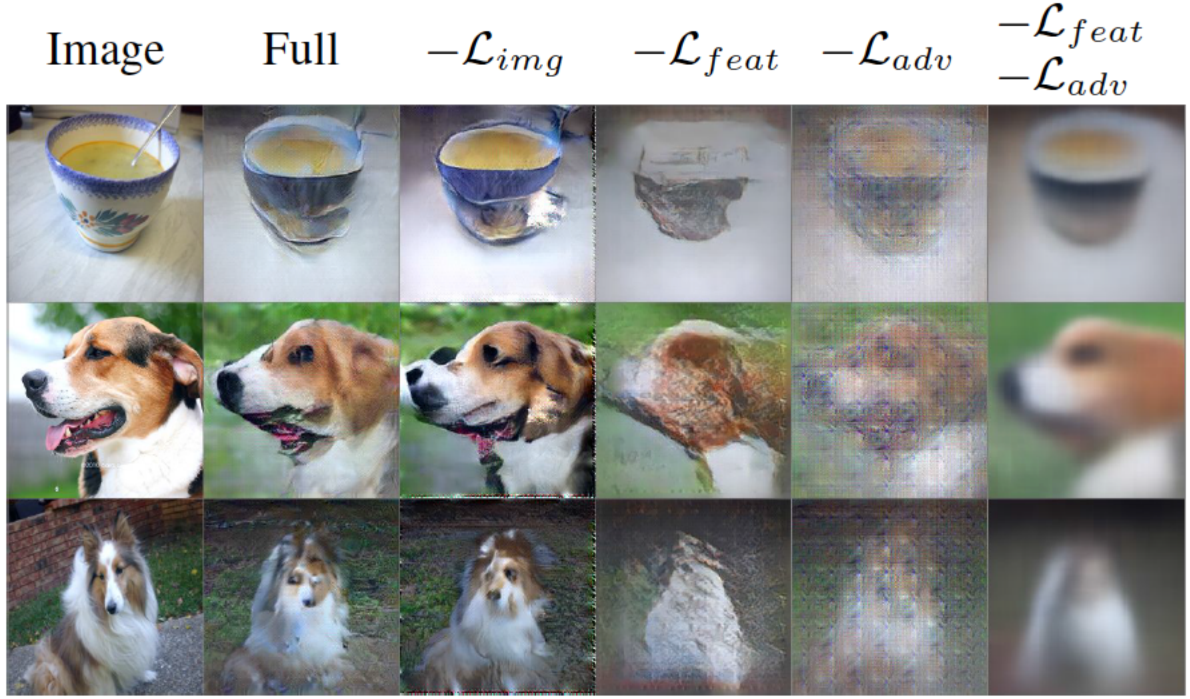


Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network

SRGAN

Ledig et al.

ECCV, 2016

arXiv:1609.04802

- Use VGG loss + general adv loss for the SR problem

- Compare 4 kinds of experiement loss:

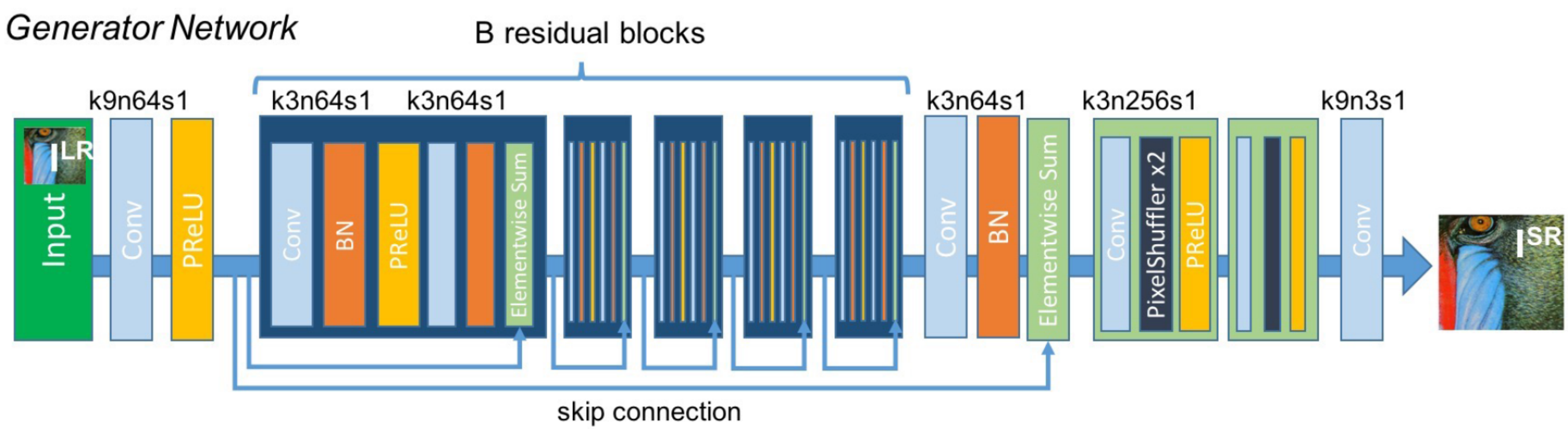
  SRResNet(MSE loss)

  SRResNet-VGG(VGG loss)

  SRGAN-MSE(MES+adv)

  SRGAN(VGG+adv)

Architecture



	SRResNet-		SRGAN-		
Set5	MSE	VGG22	MSE	VGG22	VGG54
PSNR	32.05	30.51	30.64	29.84	29.40
SSIM	0.9019	0.8803	0.8701	0.8468	0.8472
MOS	3.37	3.46	3.77	3.78	3.58
Set14					
PSNR	28.49	27.19	26.92	26.44	26.02
SSIM	0.8184	0.7807	0.7611	0.7518	0.7397
MOS	2.98	3.15*	3.43	3.57	3.72*

bicubic
(21.59dB/0.6423)

SRResNet
(23.53dB/0.7832)

SRGAN
(21.15dB/0.6868)

original

Johnson et al. ECCV 2016

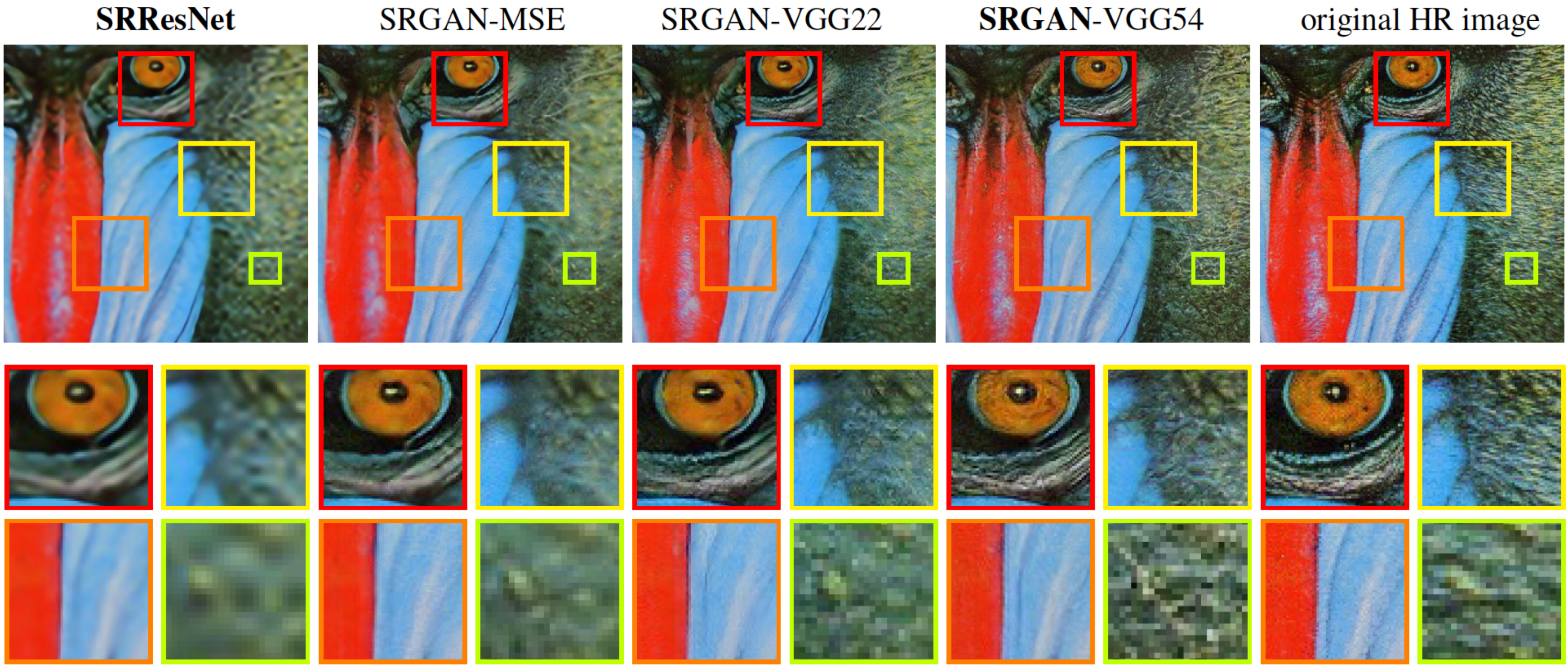

SRResNet
SRGAN-MSE
SRGAN-VGG22
SRGAN-VGG54
original HR image

Image-to-Image Translation with Conditional Adversarial Networks

Pix2Pix

Isola et al.
CVPR, 2017

arXiv:1611.07004

- General purpose supervised image to image translation
- Using conditional GAN

$$\mathcal{L}_{cGAN}(G, D) = \mathbb{E}_{x,y\sim p_{data}(x,y)}[\log D(x, y)] + \mathbb{E}_{x\sim p_{data}(x),z\sim p_z(z)}[\log(1 - D(x, G(x, z))]$$

- Beneficial to mix the GAN objective with a more traditional loss, and L1 encourages less blurring than L2

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda\mathcal{L}_{L1}(G).$$

- The generator simply learned to ignore the input noise(is an important question left open for future)

Generator Architecture: Unet

- “Such a network requires that all information flow pass through all the layers, including the bottleneck. For many image translation problems, there is a great deal of low-level information shared between the input and output, and it would be desirable to shuttle this information directly across the net.”

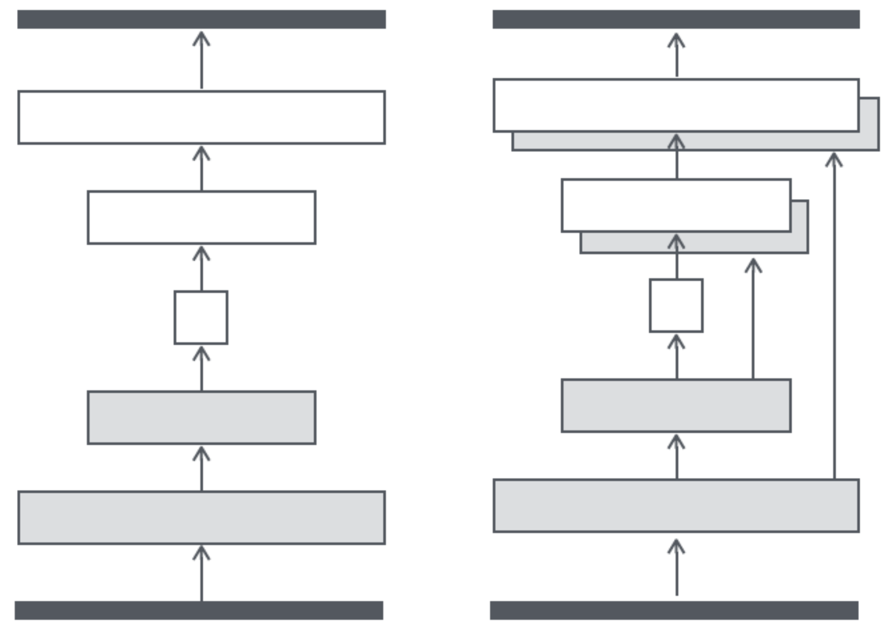


Figure 3: Two choices for the architecture of the generator. The “U-Net” [34] is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.

Discriminator Architecture: PatchGAN

- "Although L2/L1 losses fail to encourage high frequency crispness, in many cases they nonetheless accurately capture the low frequencies. For problems where this is the case, we do not need an entirely new framework to enforce correctness at the low frequencies. L1 will already do.

- This motivates restricting the GAN discriminator to only model high-frequency structure, relying on an L1 term to force low-frequency correctness. In order to model high-frequencies, it is sufficient to restrict our attention to the structure in local image patches. Therefore, we design a discriminator architecture – which we term a PatchGAN – that only penalizes structure at the scale of patches. This discriminator tries to classify if each NxN patch in an image is real or fake. We run this discriminator convolutionally across the image, averaging all responses to provide the ultimate output of D."

- “N can be much smaller than the full size of the image and still produce high quality results. This is advantageous because a smaller PatchGAN has fewer parameters, runs faster, and can be applied on arbitrarily large images.

- Such a discriminator effectively models the image as a Markov random field, assuming independence between pixels separated by more than a patch diameter. This is the common assumption in models of texture. Our PatchGAN can therefore be understood as a form of texture/style loss.”

Experiments

- Different losses
- Different architectures
- Human evaluation
- Segmentation task

Losses

Loss	**Per-pixel acc.**	**Per-class acc.**	**Class IOU**
L1	0.44	0.14	0.10
GAN	0.22	0.05	0.01
cGAN	0.61	**0.21**	**0.16**
L1+GAN	**0.64**	0.19	0.15
L1+cGAN	0.63	**0.21**	**0.16**
Ground truth	0.80	0.26	0.21

Table 1: FCN-scores for different losses, evaluated on Cityscapes labels↔photos.

Generator Architecture

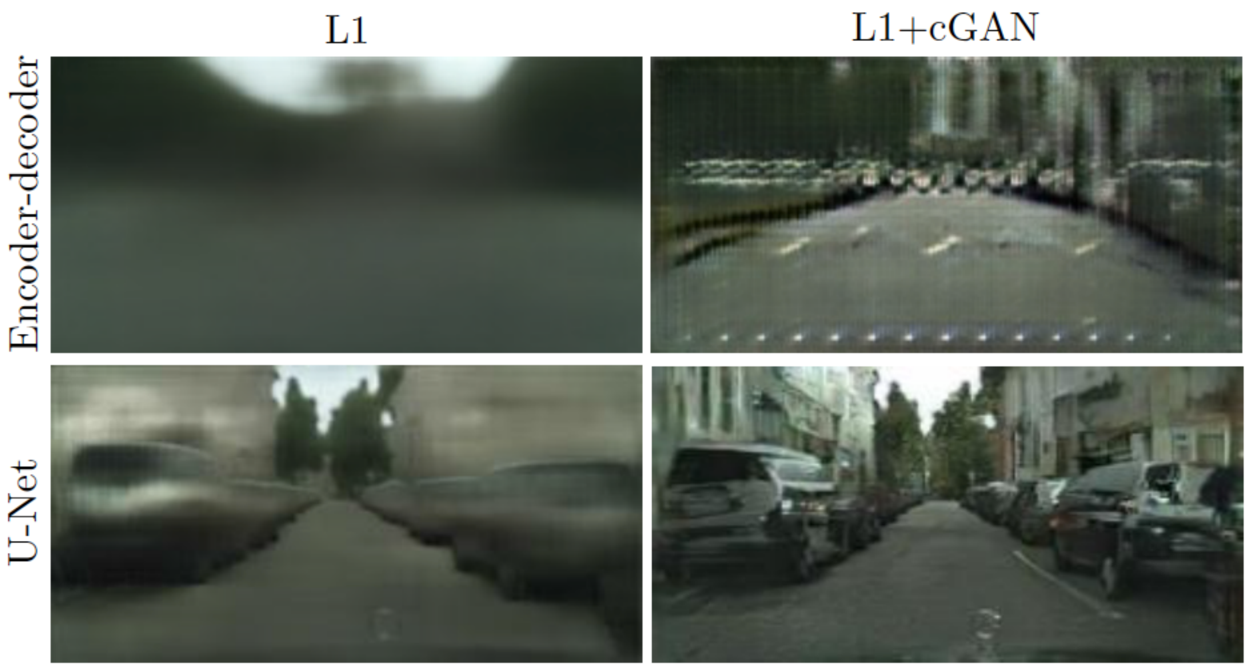


Figure 5: Adding skip connections to an encoder-decoder to create a “U-Net” results in much higher quality results.

Discriminator Architecture

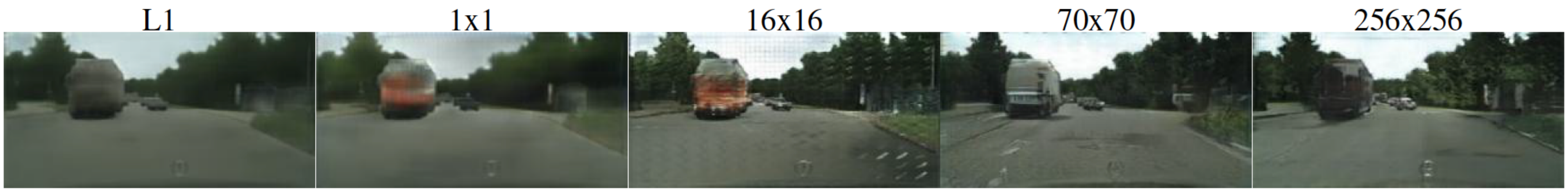


Figure 6: Patch size variations. Uncertainty in the output manifests itself differently for different loss functions. Uncertain regions become blurry and desaturated under L1. The 1x1 PixelGAN encourages greater color diversity but has no effect on spatial statistics. The 16x16 PatchGAN creates locally sharp results, but also leads to tiling artifacts beyond the scale it can observe. The 70x70 PatchGAN forces outputs that are sharp, even if incorrect, in both the spatial and spectral (coforfulness) dimensions. The full 256x256 ImageGAN produces results that are visually similar to the 70x70 PatchGAN, but somewhat lower quality according to our FCN-score metric (Table 2). Please see `https://phillipi.github.io/pix2pix/` for additional examples.

Semantic Segmentation

Loss	Per-pixel acc.	Per-class acc.	Class IOU
L1	**0.86**	**0.42**	**0.35**
cGAN	0.74	0.28	0.22
L1+cGAN	0.83	0.36	0.29

Table 5: Performance of photo→labels on cityscapes.

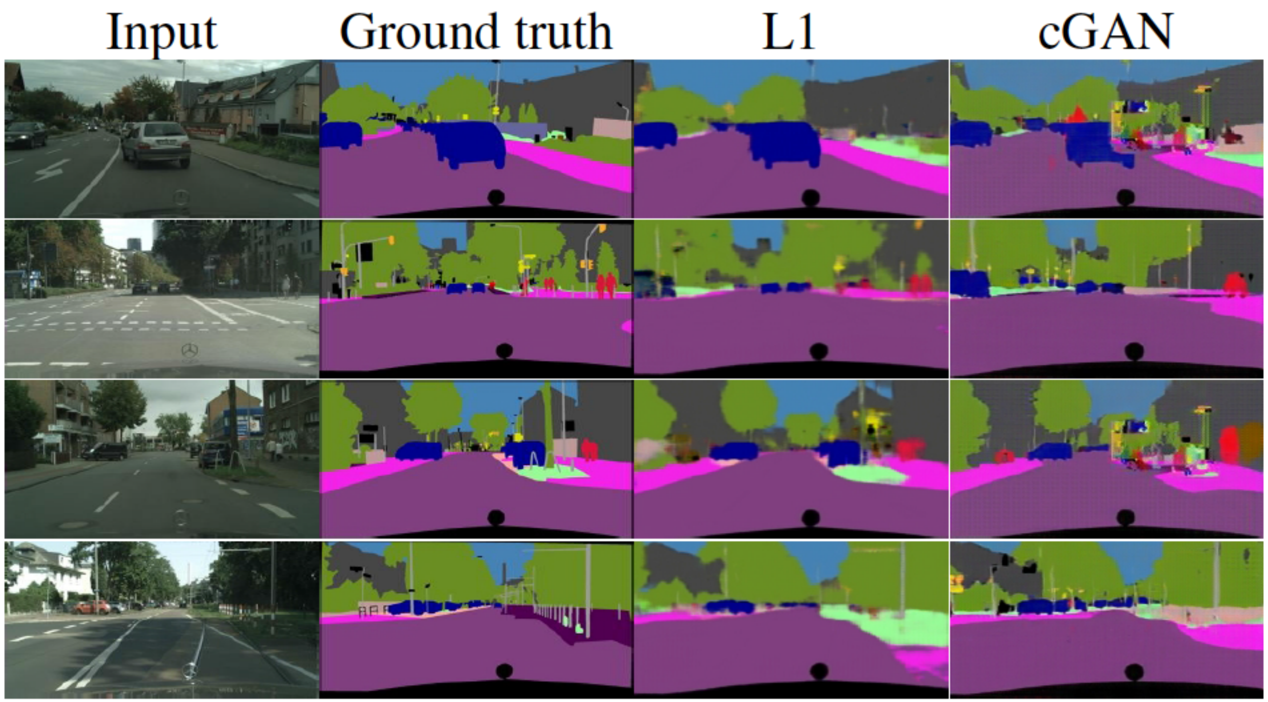


Figure 10: Applying a conditional GAN to semantic segmentation. The cGAN produces sharp images that look at glance like the ground truth, but in fact include many small, hallucinated objects.

- “Conditional GANs appear to be effective on problems where the output is highly detailed or photographic, as is common in image processing and graphics tasks.
- For vision problems, the goal (i.e. predicting output close to ground truth) may be less ambiguous than graphics tasks, and reconstruction losses like L1 are mostly sufficient.”

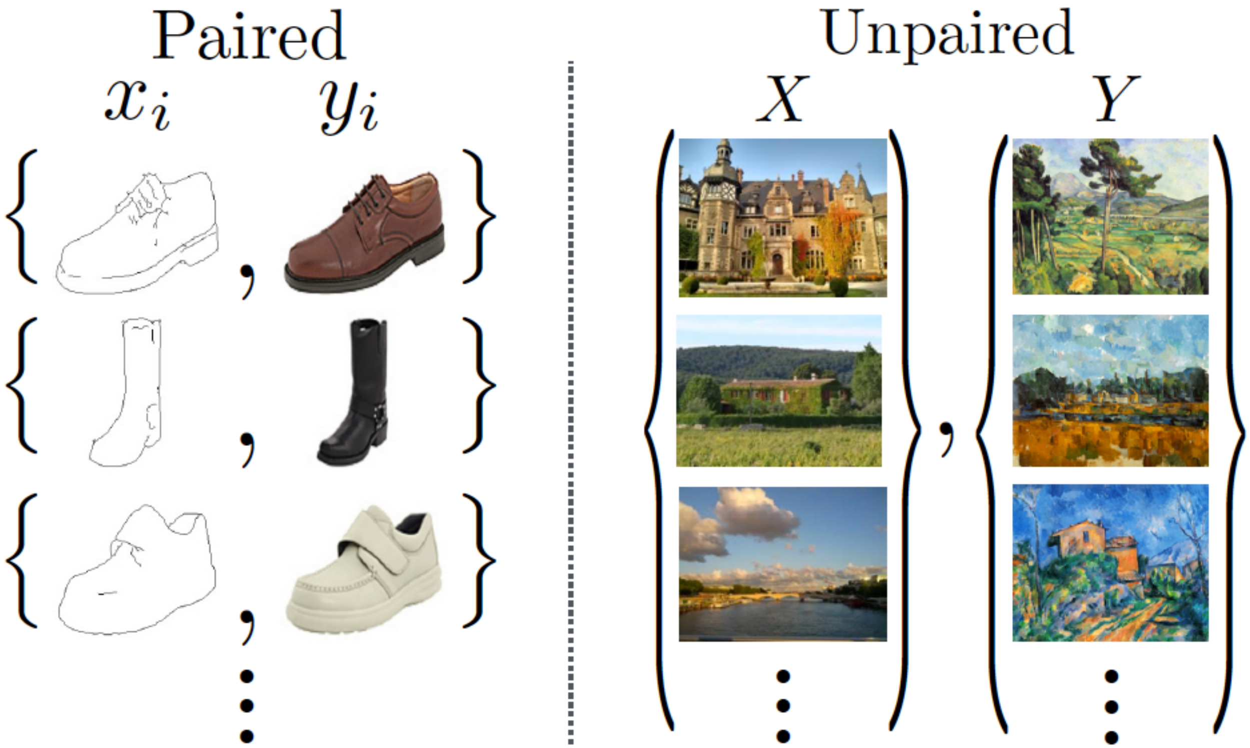
Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

CycleGAN

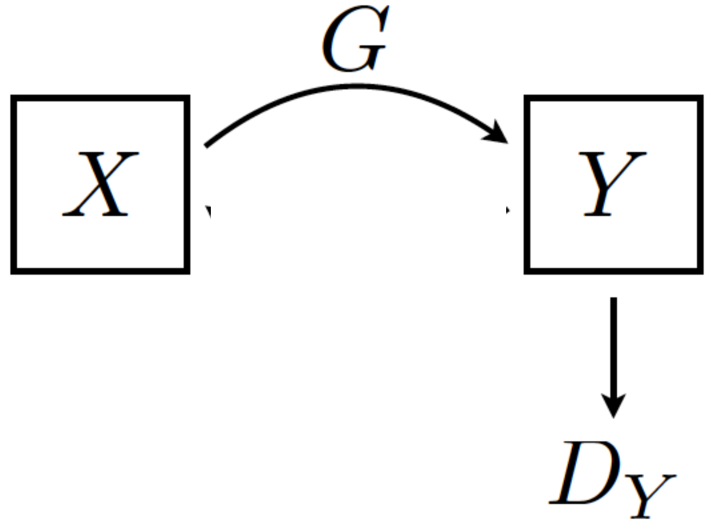
Zhu et al.

arXiv:1703.10593

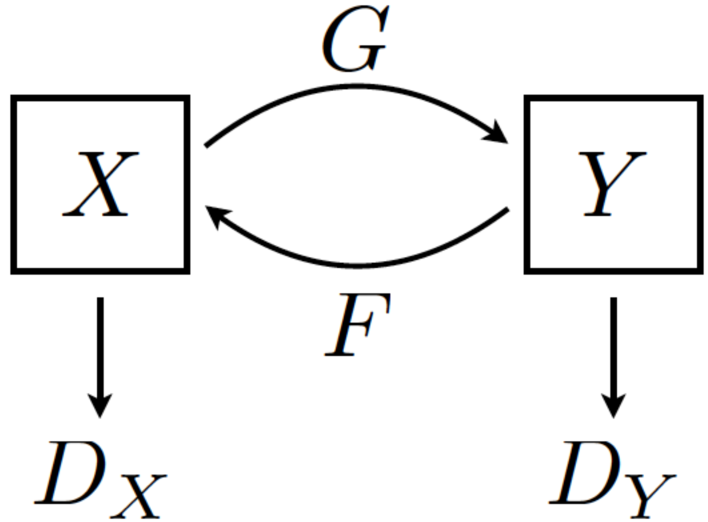
- General purpose unsupervised image to image translation
- Using cycle consistency constraint
- Achieve bi-direction translation



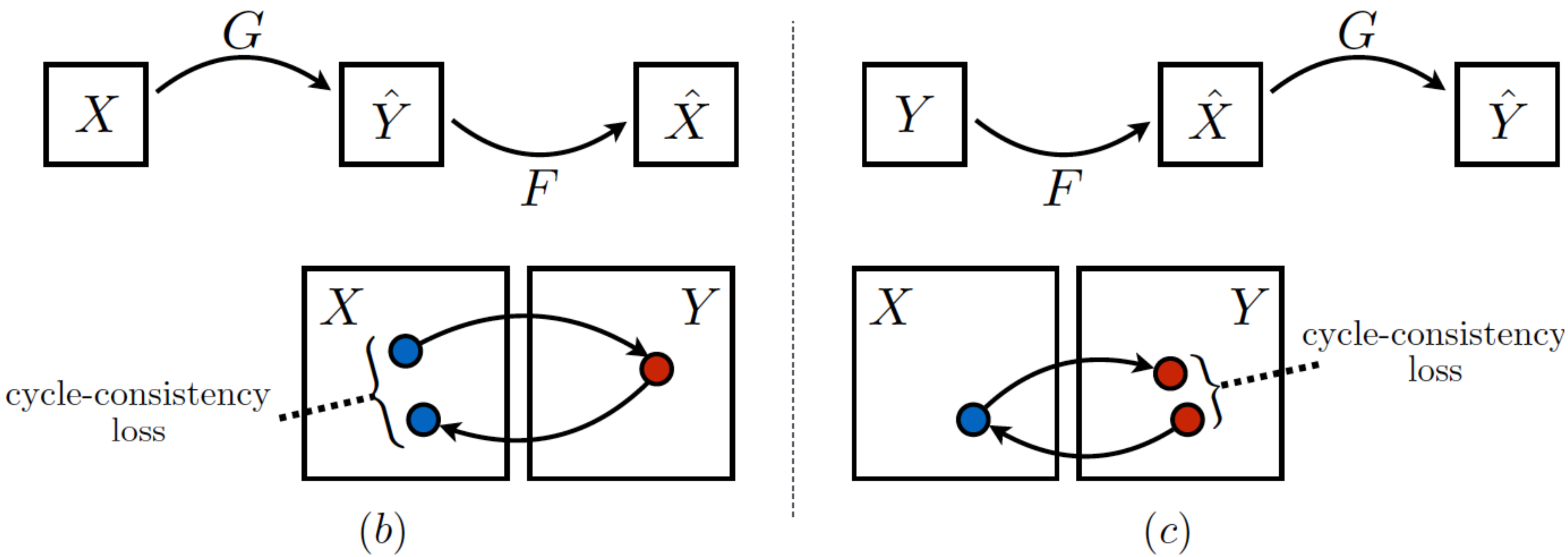
- getting paired dataset is difficult and expensive
- Assume there is some underlying relationship between the domains – for example, that they are two different renderings of the same underlying world – and seek to learn that relationship.



- General adv loss can’t constrain the input-output relationship and results in mode collapse



- General adv loss can’t constrain the input-output relationship and results in mode collapse
- Solution: cycle-consistency



- Autoencoder view: AE with a meaningful intermediate representation
- Dual learning view: DualGAN

$$
\begin{aligned}
\mathcal{L}_{\text{cyc}}(G, F) = & \mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\
& + \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1].
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{L}(G, F, D_X, D_Y) = & \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\
& + \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\
& + \lambda \mathcal{L}_{\text{cyc}}(G, F),
\end{aligned}
$$

- Training tricks:
- LSGAN loss

$$\begin{aligned}\mathcal{L}_{\text{LSGAN}}(G, D_Y, X, Y) =& \mathbb{E}_{y\sim p_{\text{data}}(y)}[(D_Y(y) - 1)^2] \\ &+ \mathbb{E}_{x\sim p_{\text{data}}(x)}[D_Y(G(x))^2],\end{aligned}$$

- Using a history of generated images(Shrivastava et al.)

Experiments

- Comparison with Other Approaches
- Analysis of the Loss Function
- Comparison with Neural Style

Comparison with Other Approaches

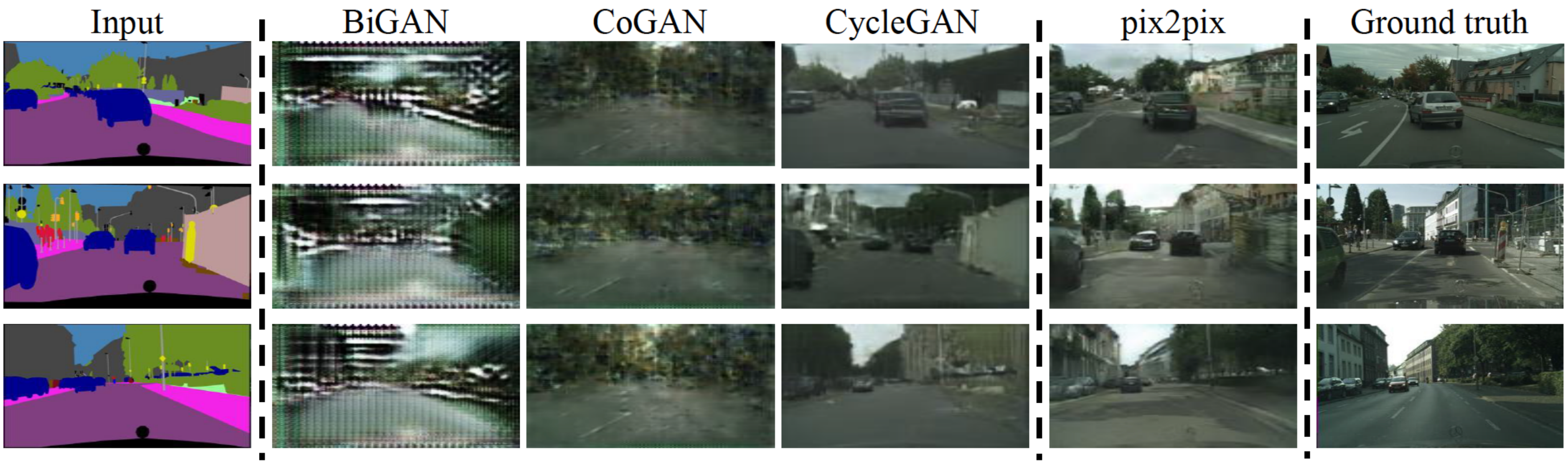


Figure 5: Different methods for mapping labels↔photos trained on cityscapes. From left to right: input, BiGAN [5, 6], CoupledGAN [27], CycleGAN (ours), pix2pix [18] trained on paired data, and ground truth.

- Unable to achieve compelling results with any other approach

Analysis of the Loss Function

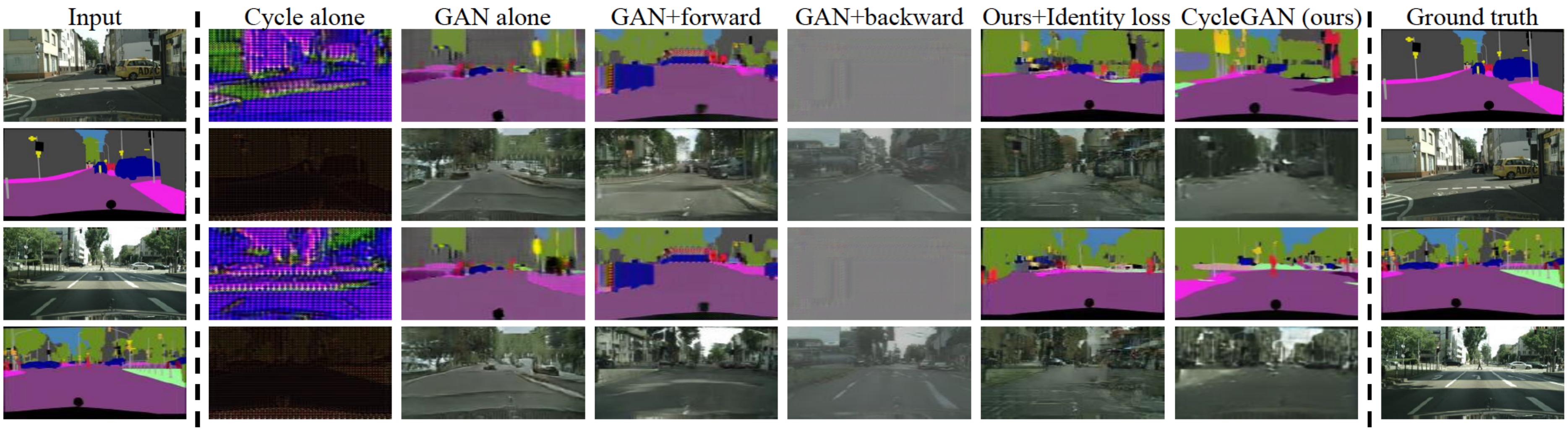
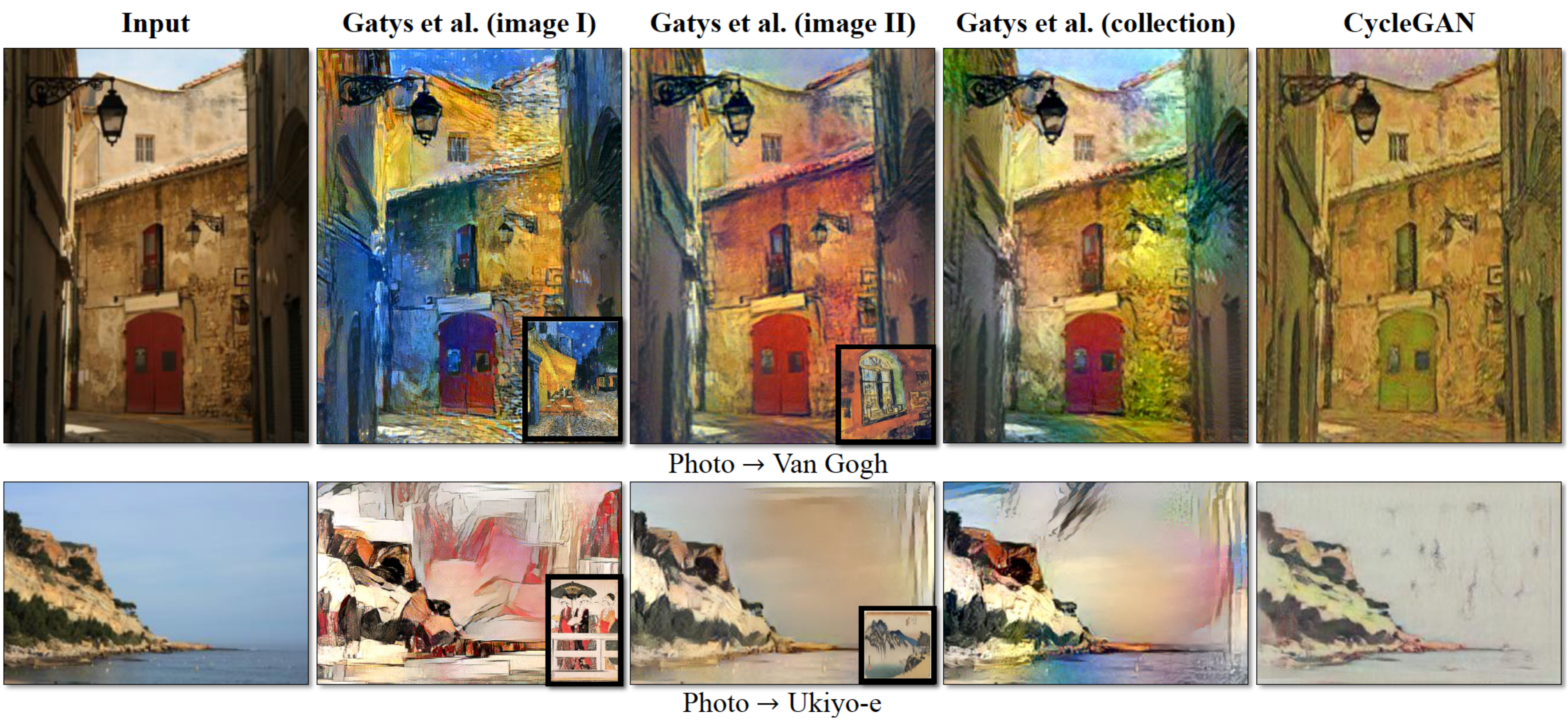


Figure 7: Different variants of our method for mapping labels↔photos trained on cityscapes. From left to right: input, cycle-consistency loss alone, adversarial loss alone, GAN + forward cycle-consistency loss ($F(G(x)) \approx x$), GAN + backward cycle-consistency loss ($G(F(y)) \approx y$), CycleGAN (our full method), and ground truth. Both *Cycle alone* and *GAN + backward* fail to produce images similar to the target domain. *GAN alone* and *GAN + forward* suffer from mode collapse, producing identical label maps regardless of the input photo.

Comparison with Neural Style



- “Handling more varied and extreme transformations, especially geometric changes, is an important problem for future work.”
- “Integrating weak or semi-supervised data may lead to substantially more powerful translators.”

Video Prediction

Deep multi-scale video prediction beyond mean square error

Mathieu et al.

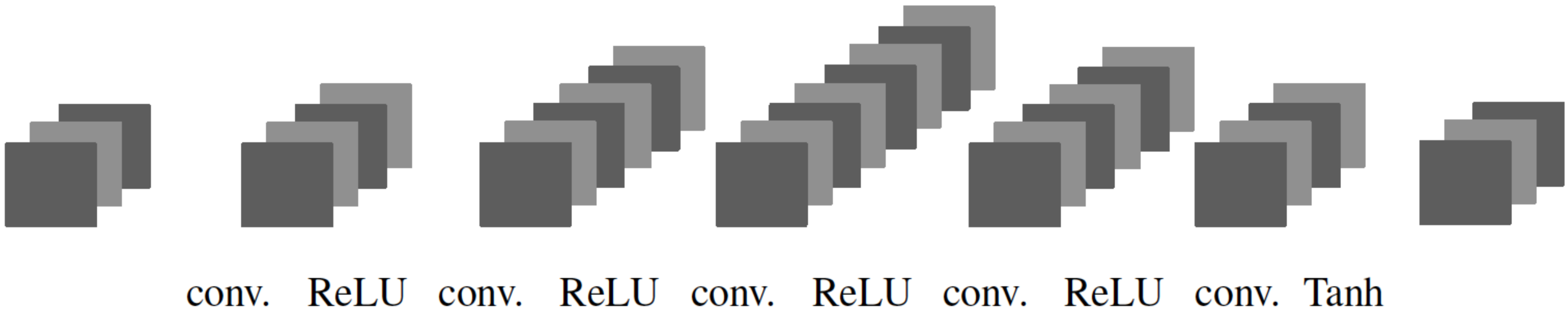
ICLR 2016

arXiv:1511.05440

Task Definition

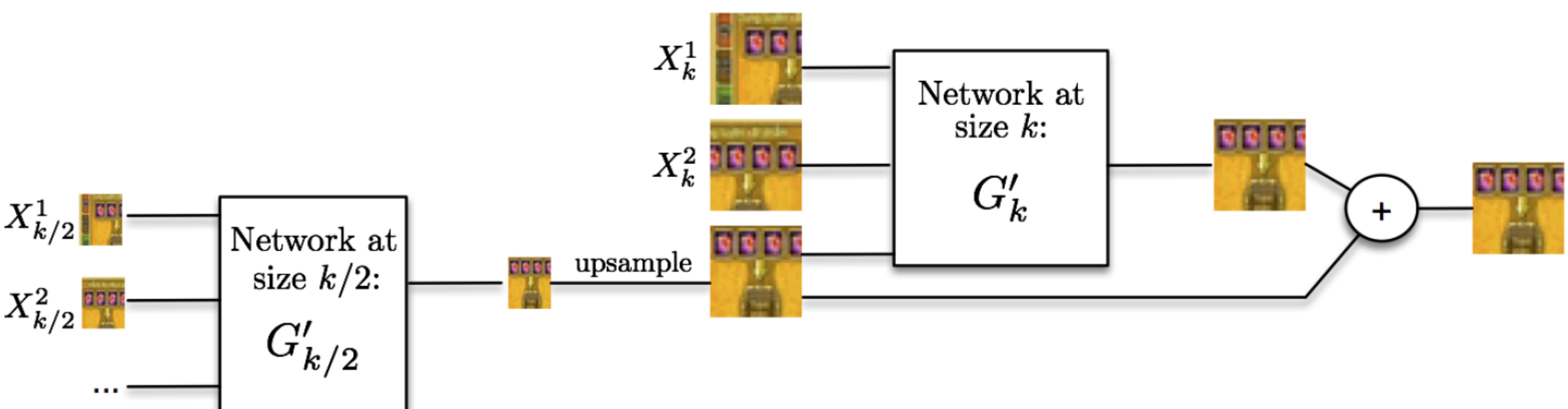
- Given fixed number of input frames, predict fixed number of output frames

Architecture



Multi-scale model tackles long-range dependency of pixels

Figure 2: Multi-scale architecture



Losses

- Conditional adv loss

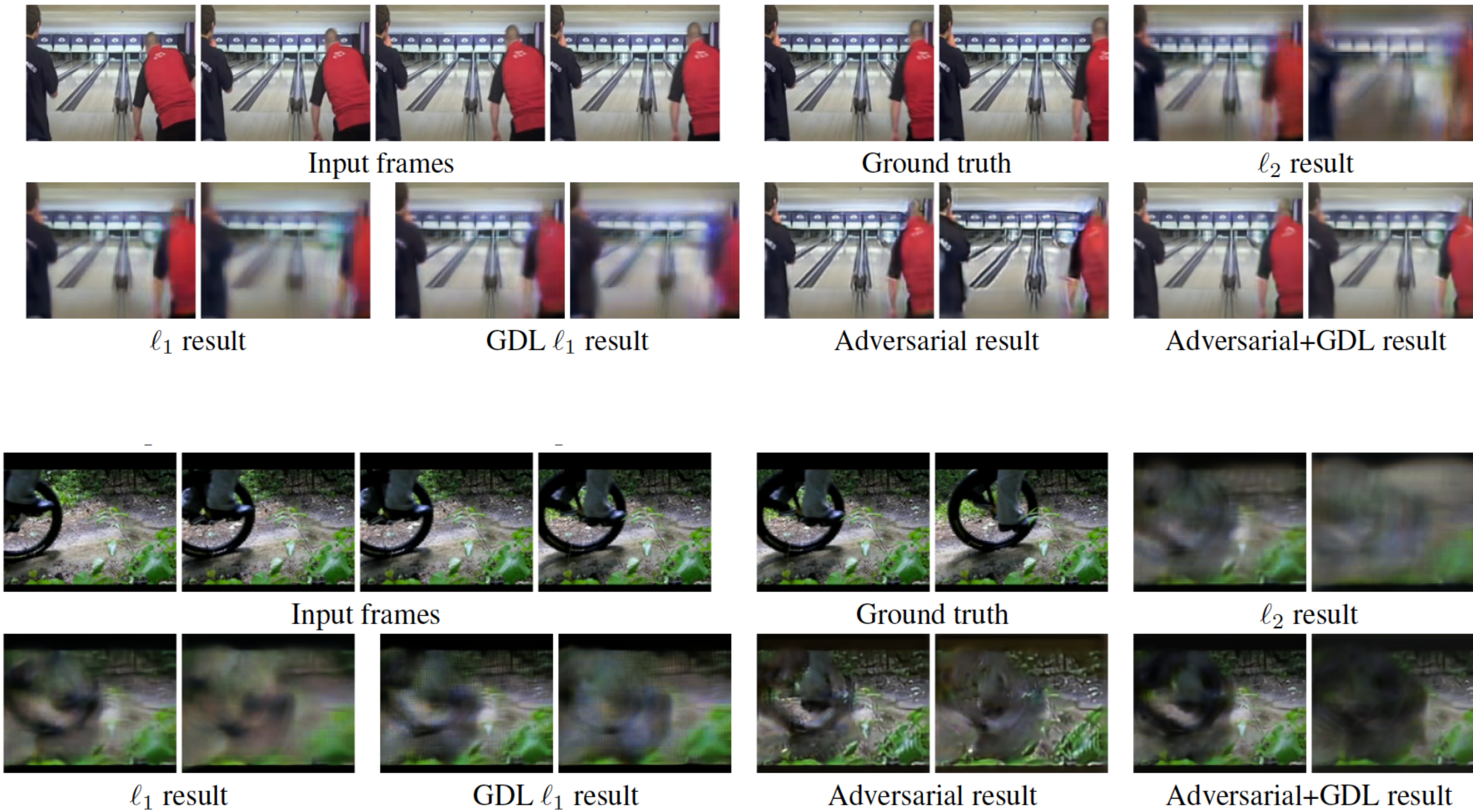
$$\mathcal{L}_{adv}^{D}(X,Y) = \sum_{k=1}^{N_{\text{scales}}} L_{bce}(D_k(X_k, Y_k), 1) + L_{bce}(D_k(X_k, G_k(X)), 0)$$

- Lp loss to stabilize adv training

- GDL(Gradient Difference Loss): sharpness

$$\mathcal{L}_{gdl}(X,Y) = L_{gdl}(\hat{Y}, Y) = \sum_{i,j} \left| |Y_{i,j} - Y_{i-1,j}| - |\hat{Y}_{i,j} - \hat{Y}_{i-1,j}| \right|^{\alpha} + \left| |Y_{i,j-1} - Y_{i,j}| - |\hat{Y}_{i,j-1} - \hat{Y}_{i,j}| \right|^{\alpha},$$

Figure 4: Results on 3 video clips from Sport1m. Training: 4 inputs, 1 output. Second output computed recursively.

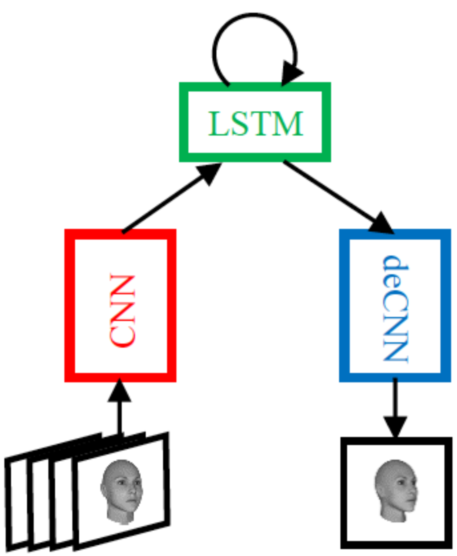


Unsupervised Learning of Visual Structure Using Predictive Generative Networks

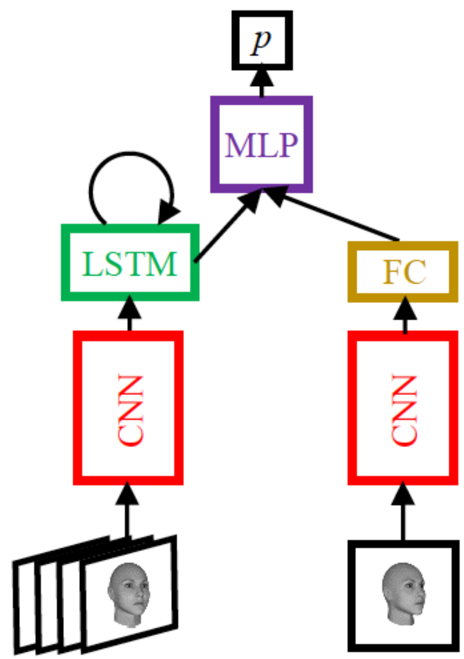
Lotter et al.
arXiv:1511.06380

Architecture & Loss

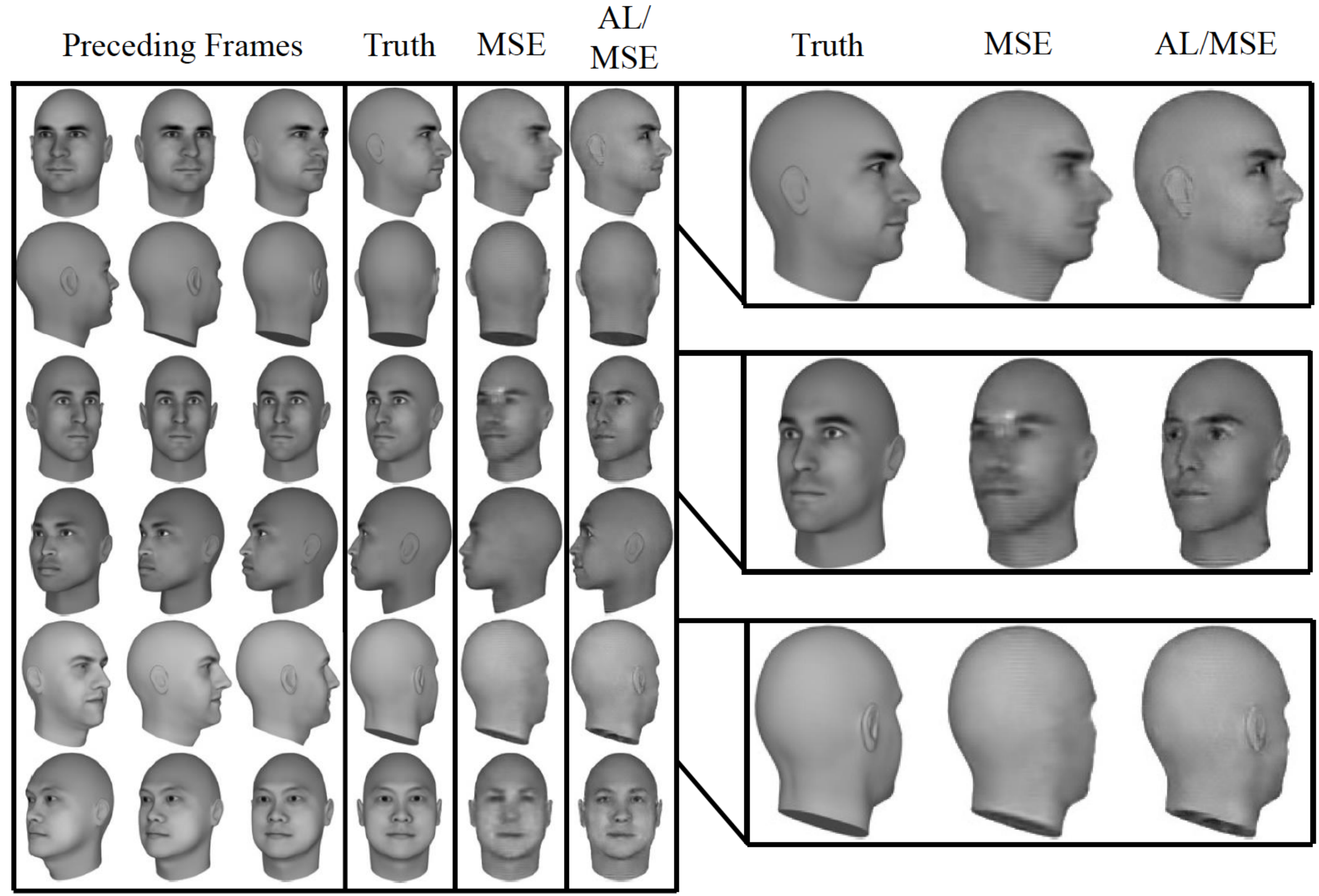
- A variable number of frames as input
- Conditional adv + MSE



Predictive Generative Network



Adversarial Discriminator


Preceding Frames
Truth
MSE
AL/
MSE
Truth
MSE
AL/MSE

- “Most notably, the AL/MSE model has learned that faces contain conspicuous eyes and ears, which are largely omitted by the MSE model.
- When the AL/MSE model does make mistakes, it’s often through generating faces that notably look realistic, but seem slightly inconsistent with the identity of the face in the preceding frames. This can be seen in the second row in the right panel of Figure 3.
- Weighting AL higher exaggerates this effect.
- One would hope that the discriminator would be able to discern if the identity changed for the proposed rotated view, but interestingly, even humans struggle with this task.”

Table 2: Decoding accuracy (r^2) of latent variables from the LSTM hidden unit representation.

Model	Angle	Speed	PC1	PC2	PC3	PC4
PGN (MSE)	0.994	0.986	0.877	0.826	0.723	0.705
PGN (AL/MSE)	0.994	0.990	0.873	0.828	0.724	0.686
Autoencoder (MSE)	0.943	0.927	0.834	0.772	0.655	0.635

Photo Editing

Generative Visual Manipulation on the Natural Image Manifold

iGAN

Zhu et al.

ECCV 2016

arXiv:1609.03552

- Common photo editing tools can achieve impressive results in the hands of an expert, but when these types of methods fail, they produce results that look nothing like a real image.

- This paper proposes to constrain the edited image on the natural image manifold by model the manifold with GAN

Natural Image Manifold

- Train DCGAN in a set of natural images
- Then all the editing can operate in the latent space

- After you have trained the GAN, you can start editing.

Photo Editing Step1

- We refer the generator of DCGAN as G
- Find the latent code of the given image, via combination of feed-forward network and optimization-based generation

$$z^* = \arg\min_{z \in \tilde{\mathbb{Z}}} \mathcal{L}(G(z), x^R).$$

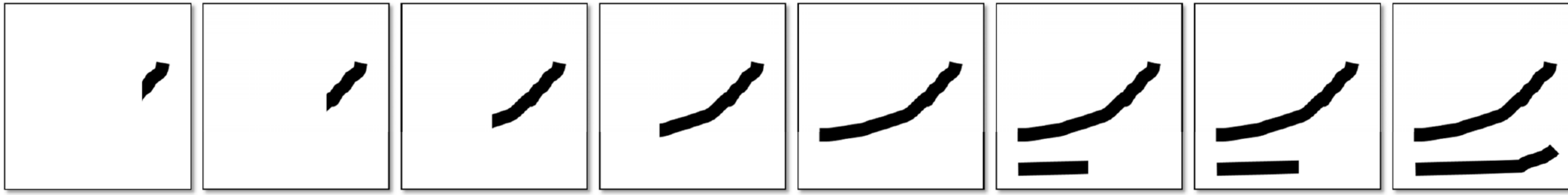
- L corresponds to a weighted combination of raw pixels and conv4 features extracted from AlexNet


Original photos
Reconstruction via Optimization
0.165 0.164 0.370 0.279 0.350 0.249 0.437 0.255 0.178 0.227
Reconstruction via Network
0.198 0.190 0.382 0.302 0.251 0.339 0.482 0.270 0.248 0.263
Reconstruction via Hybrid Method
0.133 0.141 0.298 0.218 0.160 0.204 0.318 0.185 0.183 0.190

Step2

- find new latent code satisfying user requirements and is close to original latent code via optimization-based generation

$$z^* = \arg\min_{z \in \mathbb{Z}} \Big\{ \underbrace{\sum_g \|f_g(G(z)) - v_g\|^2}_{\text{data term}} + \underbrace{\lambda_s \cdot \|z - z_0\|^2}_{\substack{\text{manifold}\\ \text{smoothness}}} + E_D \Big\}.$$



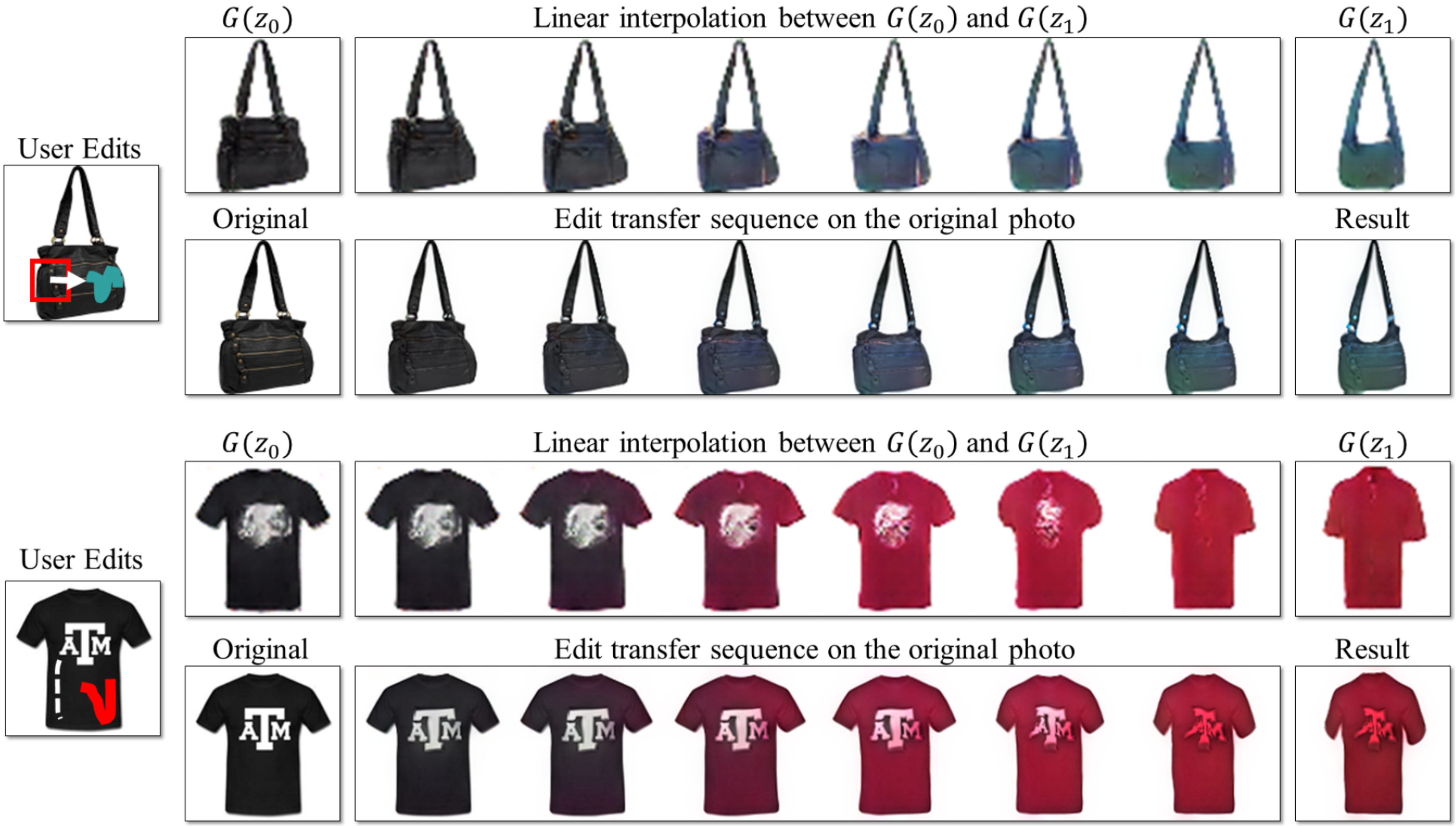
(a) User constraints v_g at different update steps

$G(z_0)$ (b) Updated images according to user edits $G(z_1)$

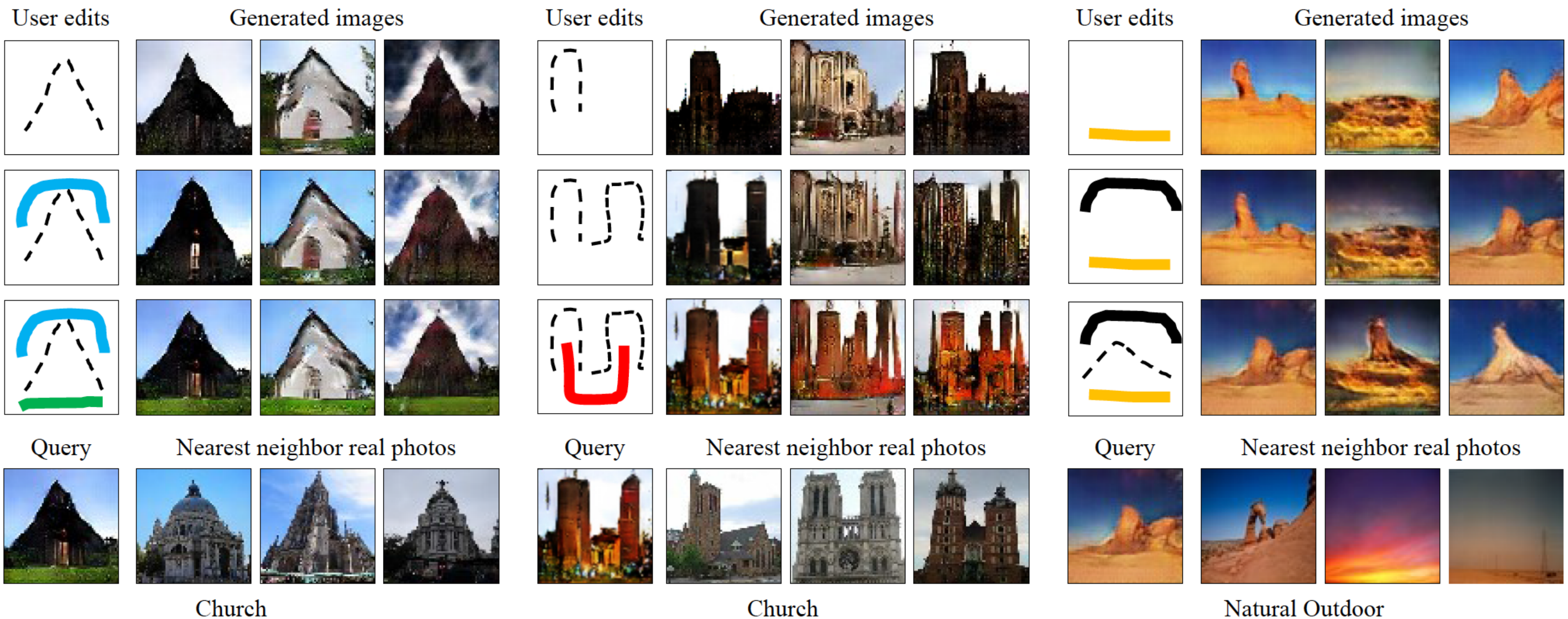
(c) Linear interpolation between $G(z_0)$ and $G(z_1)$

Step3

- Edit transfer: apply the same adjustment to the original image by optical flow method with interpolation in the latent space between z0 and z1



User Edits
$G(z_0)$
Linear interpolation between $G(z_0)$ and $G(z_1)$
$G(z_1)$
Original
Edit transfer sequence on the original photo
Result
User Edits
$G(z_0)$
Linear interpolation between $G(z_0)$ and $G(z_1)$
$G(z_1)$
Original
Edit transfer sequence on the original photo
Result



- Start editing from a white board

Thanks!
Related Works for Further Reading

- Gatys et al. A Neural Algorithm of Artistic Style
- Johnson et al. Perceptual Losses for Real-Time Style Transfer and Super-Resolution
- CGAN
- LAPGAN
- VAE-GAN
- DualGAN
- IAN