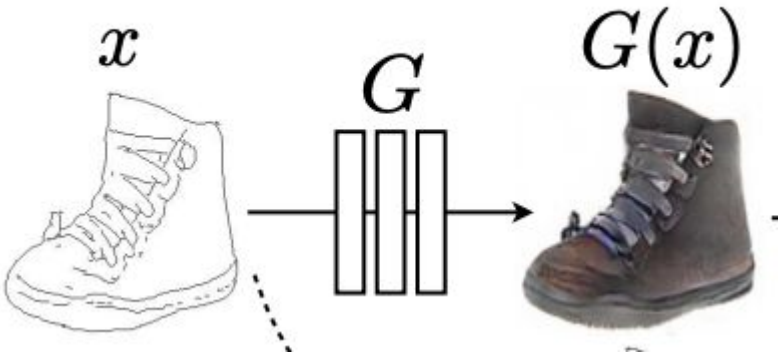
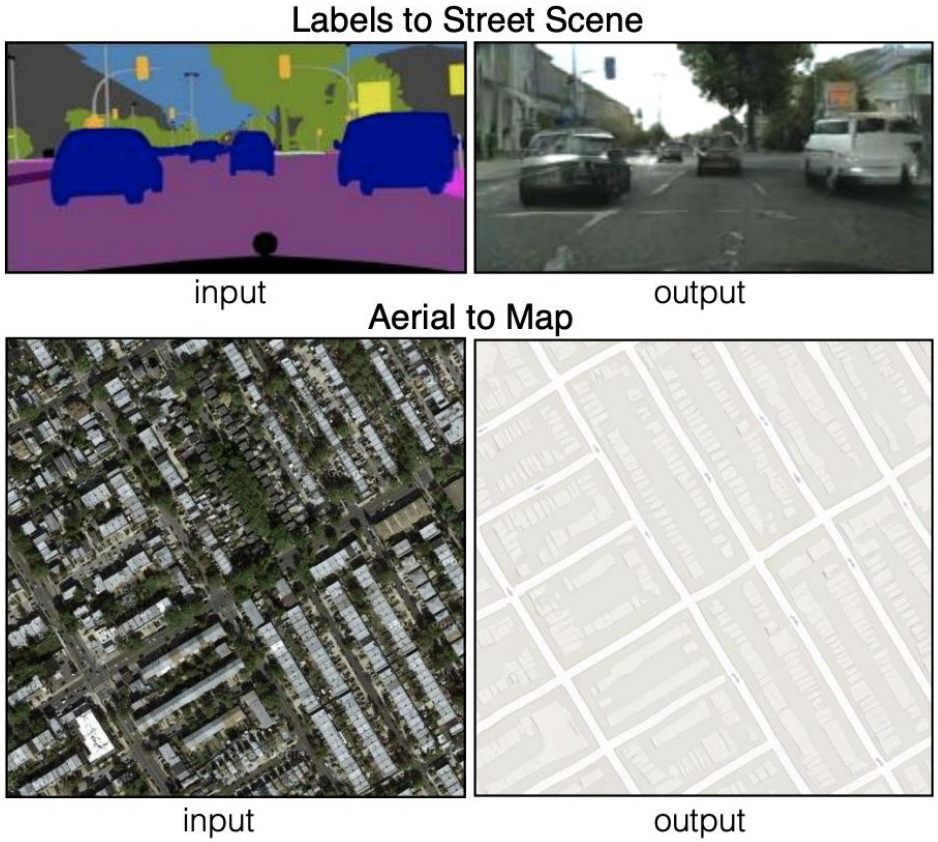
# Conditional Generation Models

# Supervised Image Translation

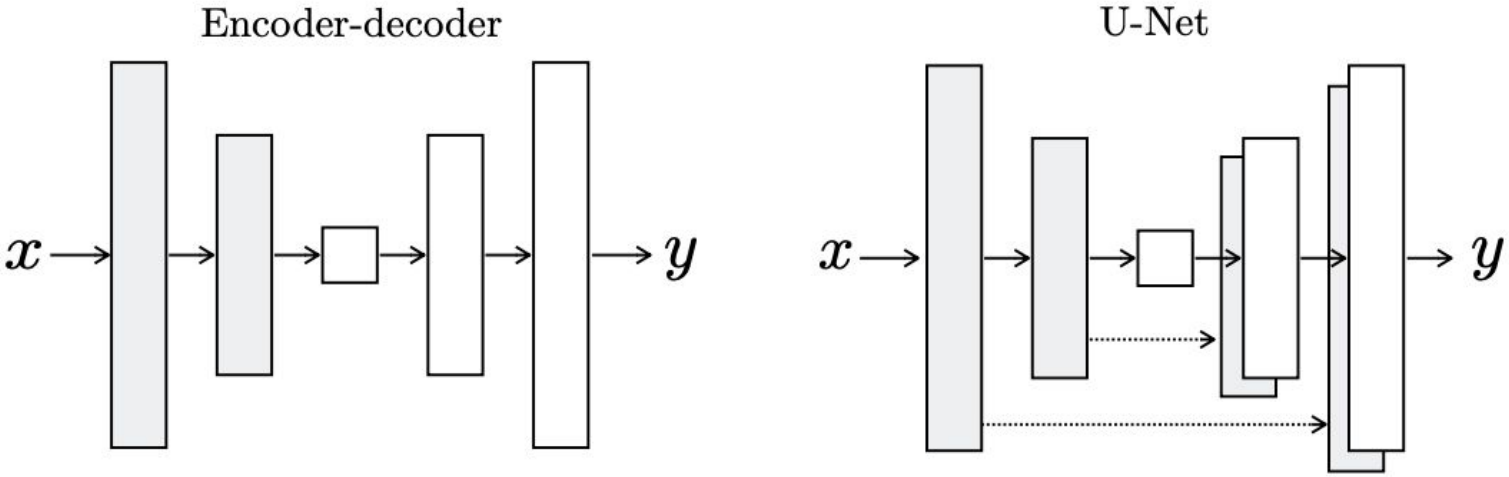


# Image Translation by Regression

Lots of early work….

Idea: Regress input to output image to minimize an L1/L2 loss



# Regression losses

Train generator G so that estimate is closest to output

Can use L1 or L2 losses.

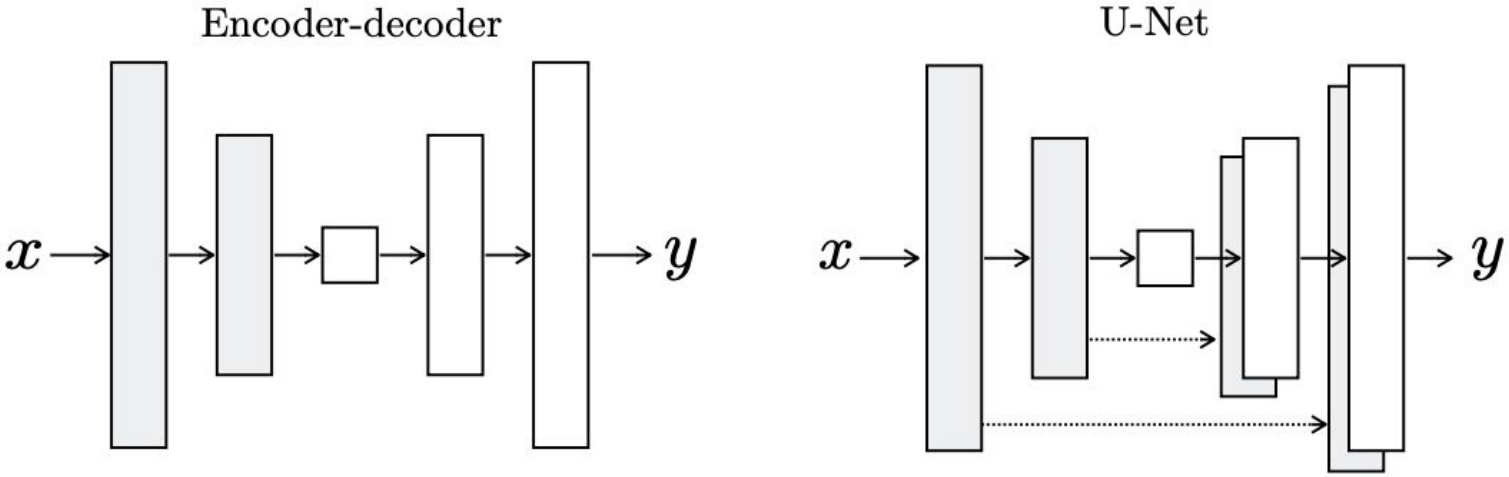
L2 is more blurry

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

# Fully Convolutional Architectures

Encoder-Decoder: reduce image size and the upscale

U-Net: Train skip connections so that resolution is not lost



# Weakness of L1/L2 losses

L1/L2 losses work great when mapping is many-to-one

They do not work when mapping is one-to-many

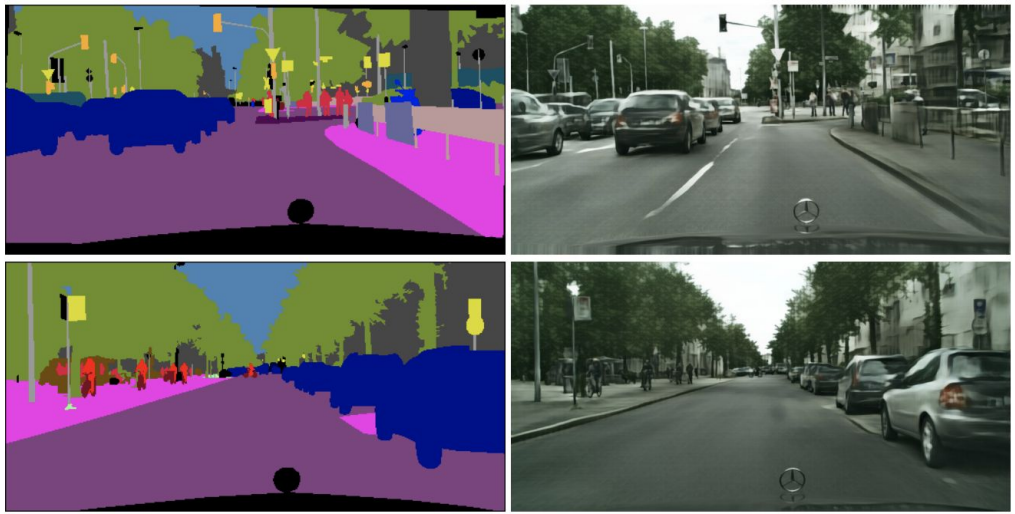
Possible tricks: predict multiple outputs and choose the best

Need a better solution

# Perceptual Loss

Qifeng Chen, Vladlen Koltun, ICCV'17

Idea: Use a perceptual loss, predict multiple outputs



(a) Input semantic layouts (b) Synthesized images

# Perceptual Loss

The perceptual loss is more sensitive to content

Content varies less than the colors

Weakness: only synthesize a single output

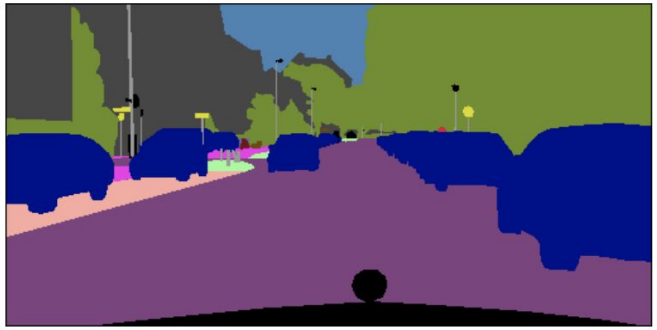
$$\mathcal{L}_{I,L}(\theta) = \sum_{l} \lambda_l \|\Phi_l(I) - \Phi_l(g(L;\theta))\|_1.$$

# Synthesizing Diverse Images

Generate multiple output image

$$\min_{u} \sum_{l} \lambda_l \|\Phi_l(I) - \Phi_l(g_u(L;\theta))\|_1.$$

# Results



Semantic layout

Our result

# Adversarial Losses

Deepak Pathak Philipp Krahenbuhl Jeff Donahue Trevor Darrell Alexei A. Efros

Idea: Use adversarial loss to ensure output looks realistic

(a) Input context (b) Human artist (c) Context Encoder ($L2$ loss) (d) Context Encoder ($L2$ + Adversarial loss)

# GAN loss

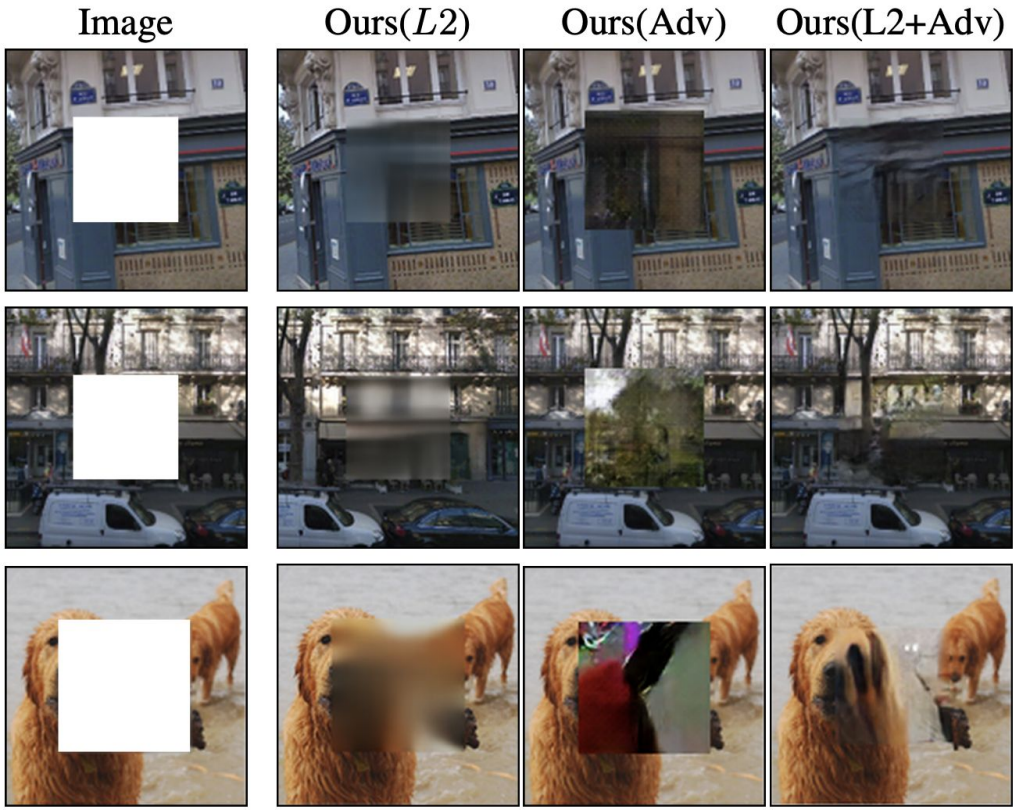
Add GAN loss to L1 reconstruction loss

Given multiple solutions, choose the one that is most relistic

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

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

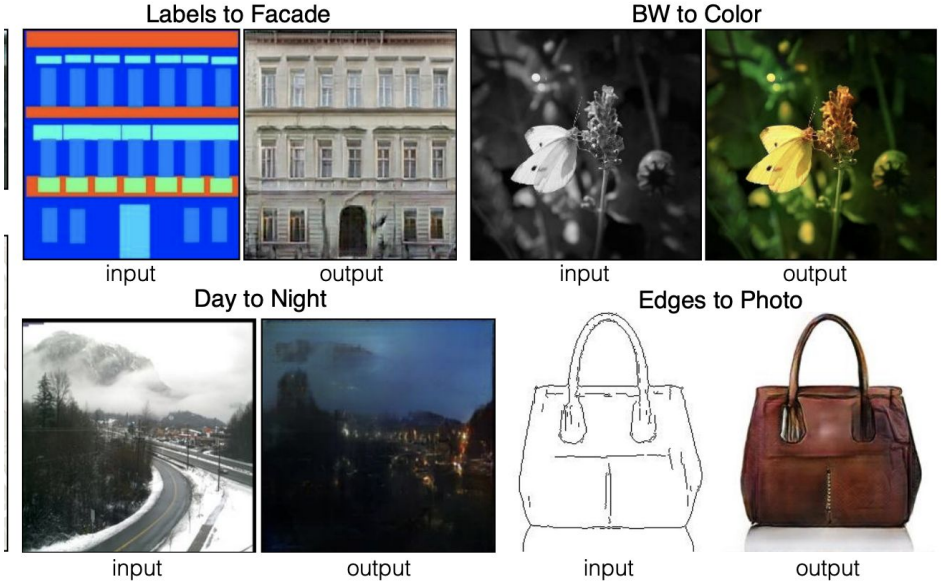
# Results



# Pix2Pix

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, Alexei A. Efros, CVPR'17

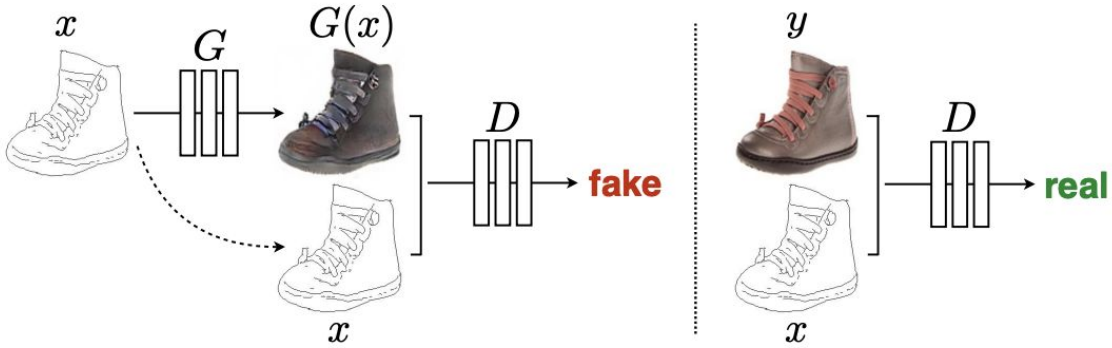
Ideas: train GANs only on patches, use cGAN



# Conditional GAN

Discriminator takes both input and output (rather than just output)

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

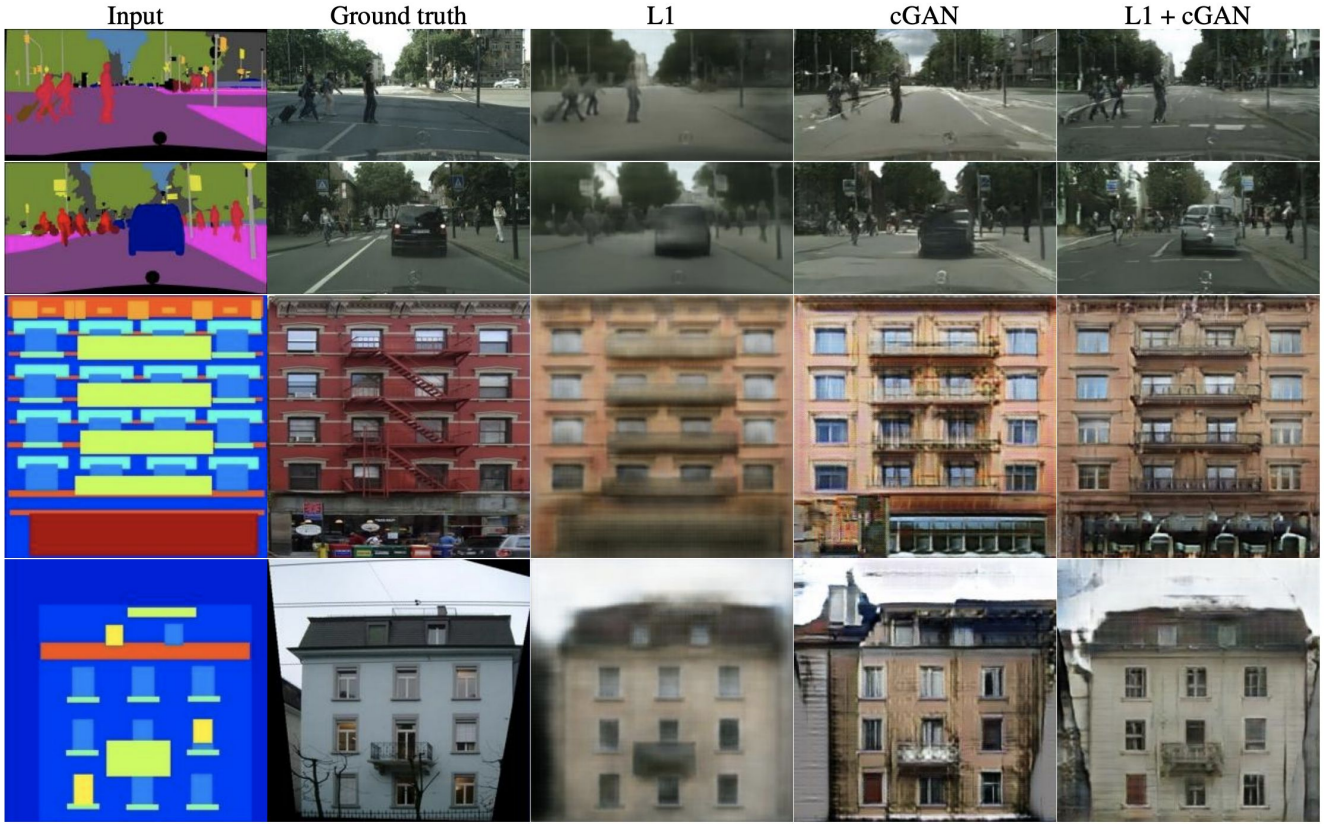


# PatchGAN Architecture

Hard to train GANs at high-resolution

Discriminator only looks at 64X64 pixels at a time

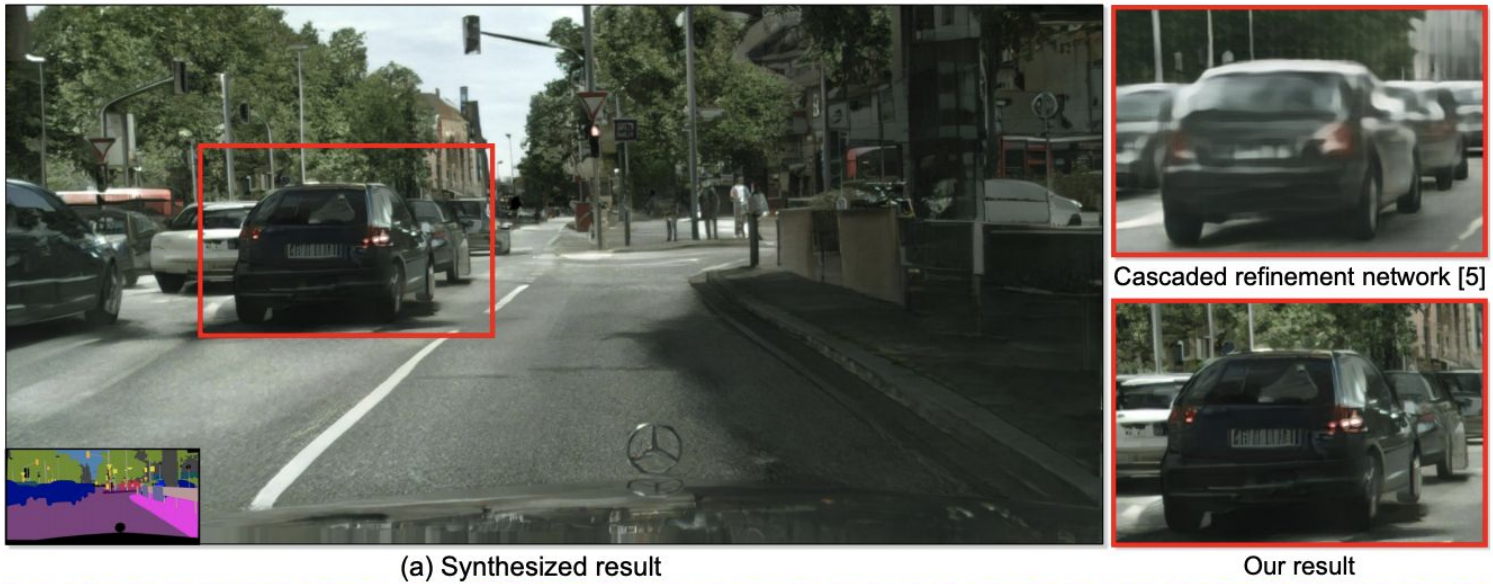
# Amazing Results



# Pix2Pix-HD

Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, Bryan Catanzaro, CVPR'18

Idea: Improve architecture to scale up to 1024X2048 resolution

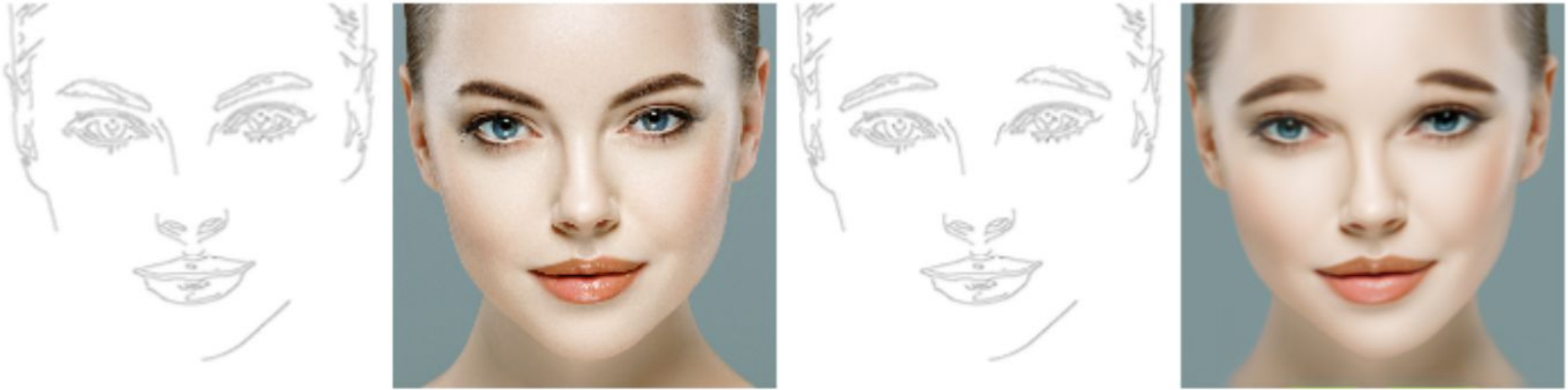


# Learning from a Single Image

Work with Yael Vinker, Eliahu Horwitz and Nir Zabari, improved version with

Input: just a single image pair

# Objective: Allow Fine Image Edits



# Challenge: Just a Single Training Image

Main issue is that there is only a single training pair

How can we predict anything when we have seen no variation?

Idea: extend a single image pair to a huge the training set by augmentation!

# Thin Plate Spline

f(x,y) =

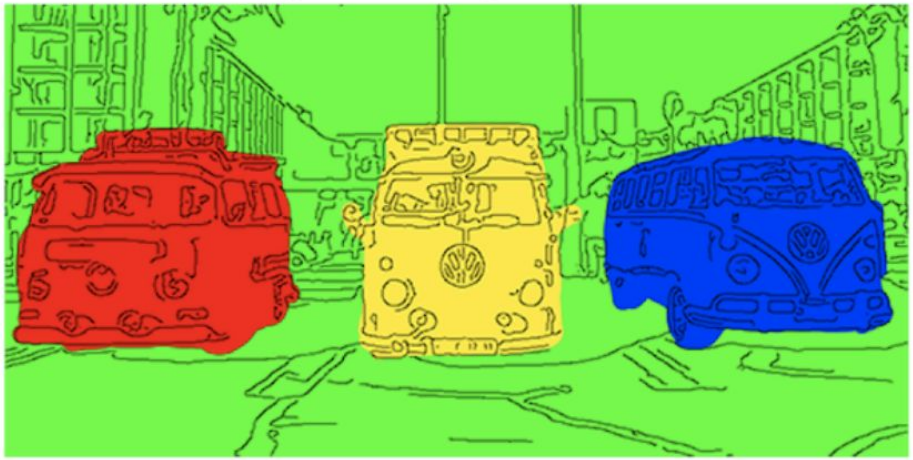
$$a_1 + a_2 x + a_3 y + \sum_{i=1}^{n} w_i U\left(|P_i - (x, y)|\right)$$

*U(r) = $r^2$ log r*

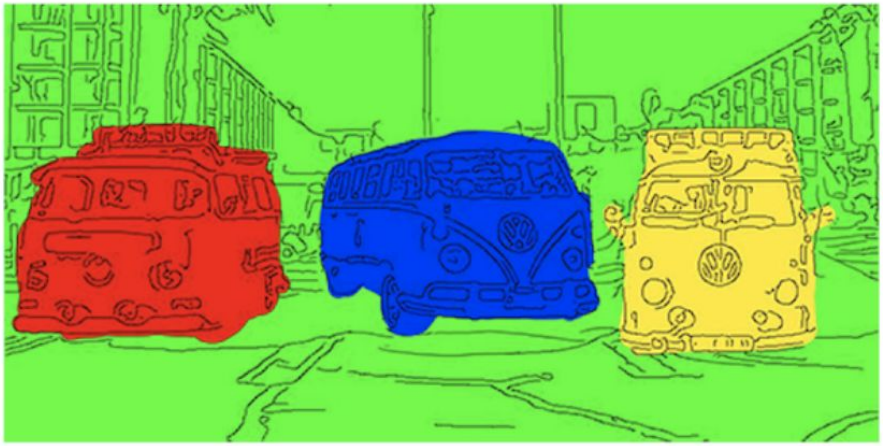
$$E_{\text{tps,smooth}}(f) = \sum_{i=1}^{K} \|y_i - f(x_i)\|^2 + \lambda \iint \left[ \left( \frac{\partial^2 f}{\partial x_1^2} \right)^2 + 2 \left( \frac{\partial^2 f}{\partial x_1 \partial x_2} \right)^2 + \left( \frac{\partial^2 f}{\partial x_2^2} \right)^2 \right] \mathrm{d}x_1 \, \mathrm{d}x_2$$

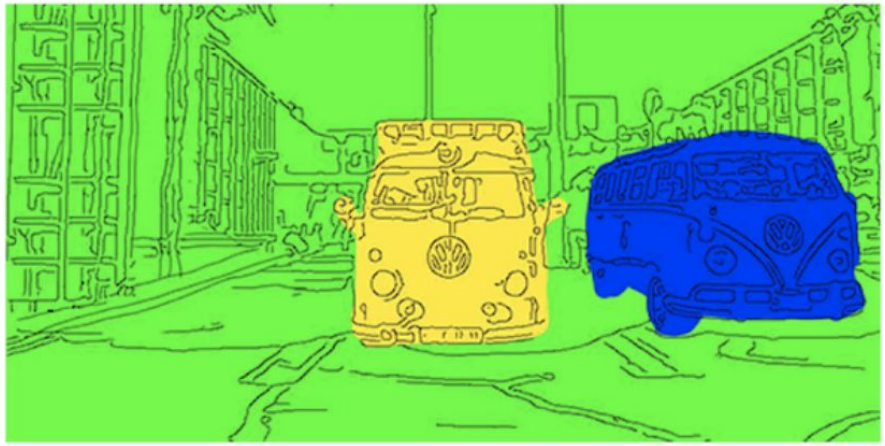
# Results

Input Image Pair



Input

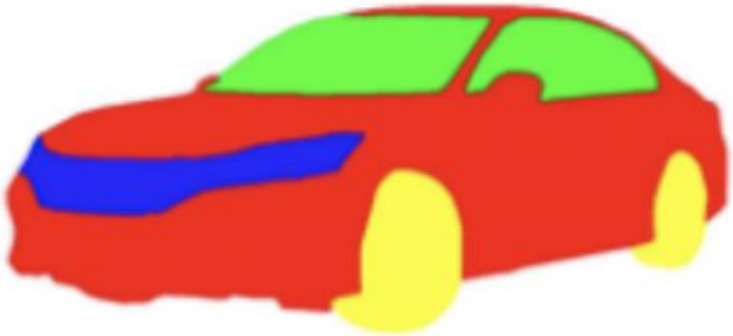


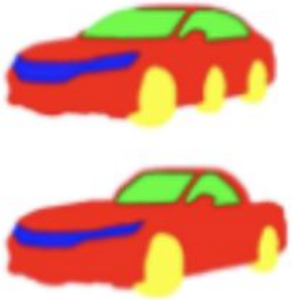


Output

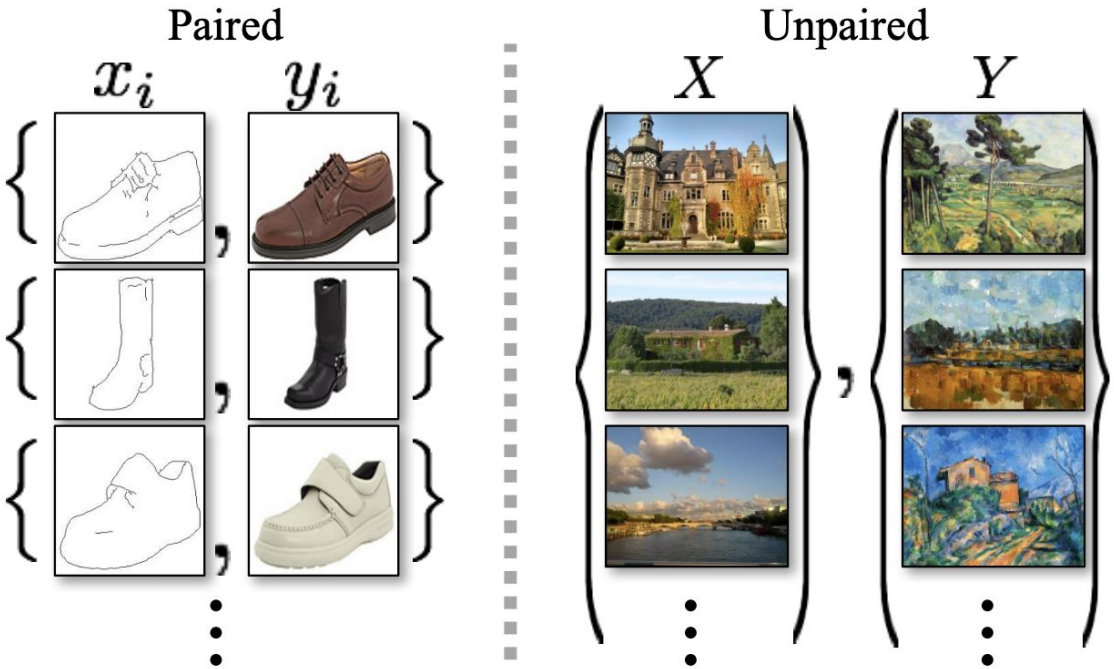
# Training

seg. to image



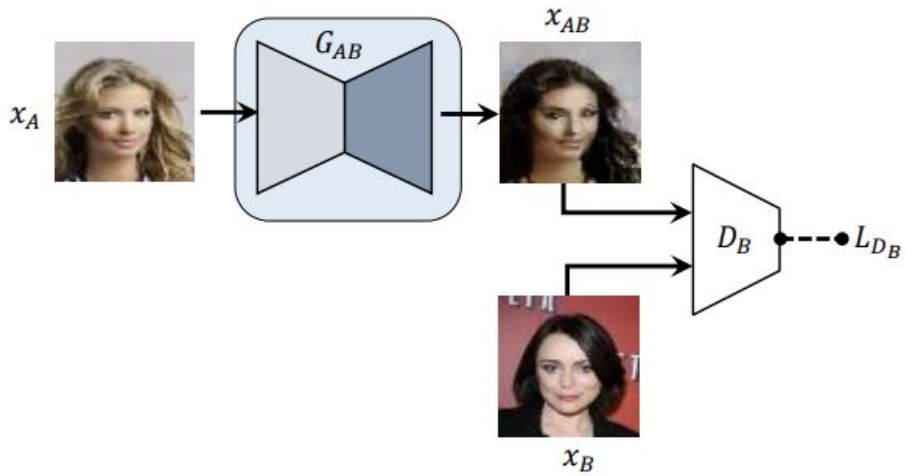


# What happens when no supervision exists?



# Adversarial Distribution Matching

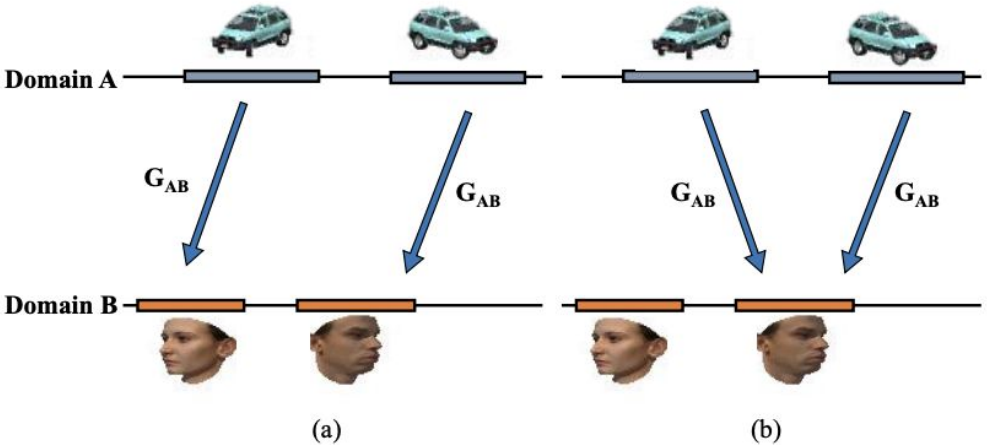
Train generator s.t. no discriminator can differentiate between G(x) and y



# Constraint not strong enough

GAN training is hard

A common failure mode is illustrated below

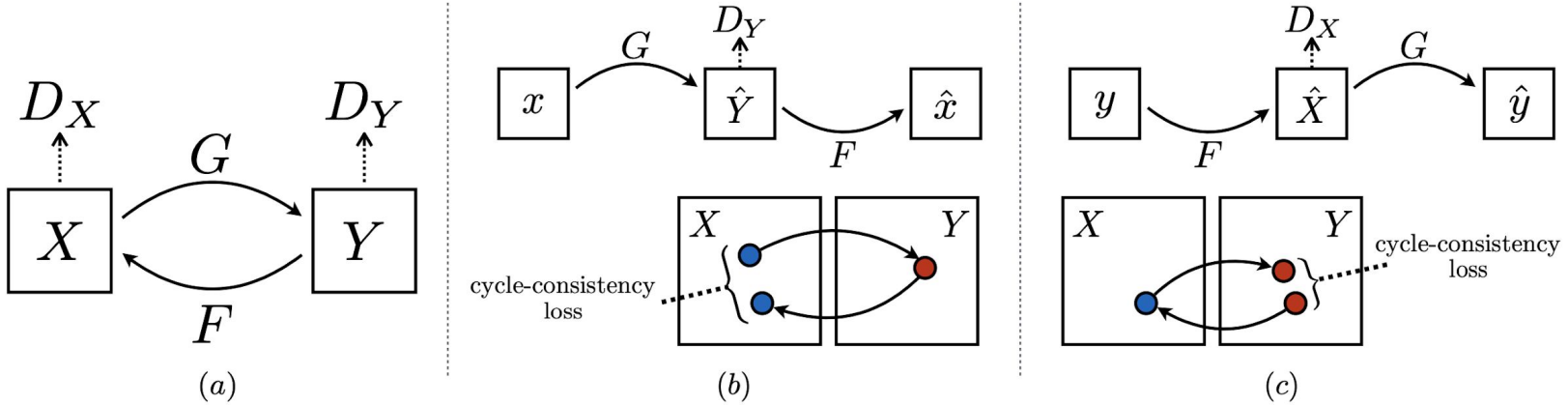


# Cycle-Constraint

Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros, ICCV'17

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, Jiwon Kim, ICML'17

Idea: images mapped from X to Y and back to X should be unchanged

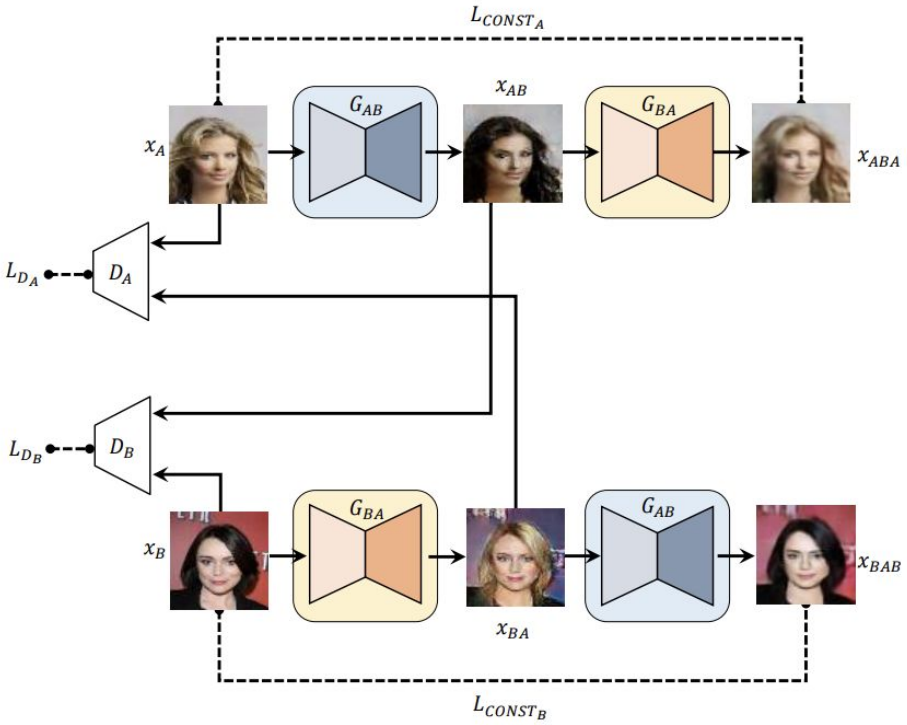


# Cycle and DiscoGAN

Train generators from X to Y and from Y to X

Train discriminators for X and Y

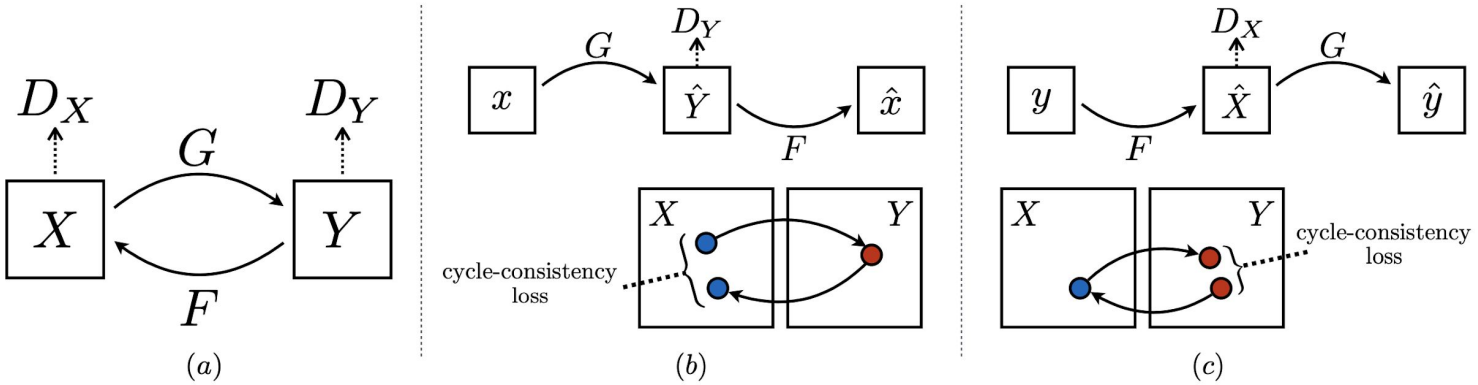
Mapped images from X to Y should look like Y images (and vice versa)



# Cycle-constraint

Images mapped from X to Y and back to Y should be unchanged

$$L_{CONST_A} = d(\mathbf{G}_{BA} \circ \mathbf{G}_{AB}(x_A), x_A)$$



# DiscoGAN and CycleGAN

Very similar ideas

CycleGAN uses a patch-discriminator, optimized for texture and style transfer

DiscoGAN looks at the entire image, but is lower resolution

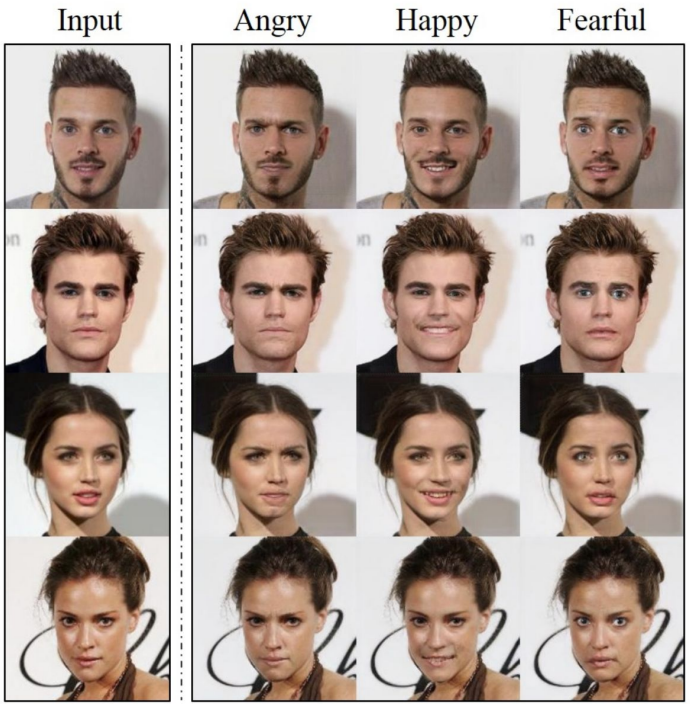
# StarGAN: Extenstion to many domains

- Although CycleGAN operated on two domains, generalization is simple

$$\mathcal{L}_{adv} = \mathbb{E}_x \left[\log D_{src}(x)\right] + \mathbb{E}_{x,c}[\log\left(1 - D_{src}(G(x,c))\right)],$$

$$\mathcal{L}_{cls}^{f} = \mathbb{E}_{x,c}[-\log D_{cls}(c|G(x,c))].$$

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

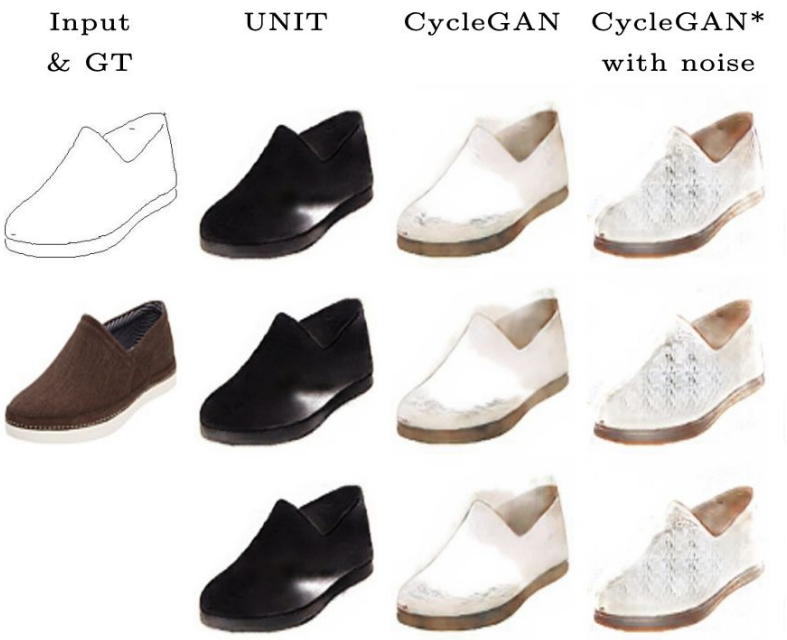


# Limitations of previous models

Previous models enforced the cycle constraint

Same as enforcing one-to-one mappings

Not a good assumption!



# Latent-space Disentanglement

- Another direction tackles attribute disentanglment
- Formulation here is by Gabbay and Hoshen, 2020
- A datum is formed by a set of attributes e.g.
  - Face: face_id, camera pose, expression, background
  - Image: color, shape, object type, object pose
  - Speech: linguistic content, speaker id, prosidy
- Disentanglement: recovering the attributes given the data

# Image Formation Model

- Image formed by class ***y*** and other attribues
- Denote the attributes correlated to the class $\boldsymbol{a_s}$
- Denote the attributes uncorrelated to the class $\boldsymbol{a_c}$

$$x_i = G^*(y_i, a_i^s, a_i^c)$$

# Image Translation

- In image translation, we want to:
  - Keep the uncorrelated attributes
  - Switch the class and class-correlated attributes

$$x_{ij} = G^*(y_i, a_i^s, a_j^c)$$

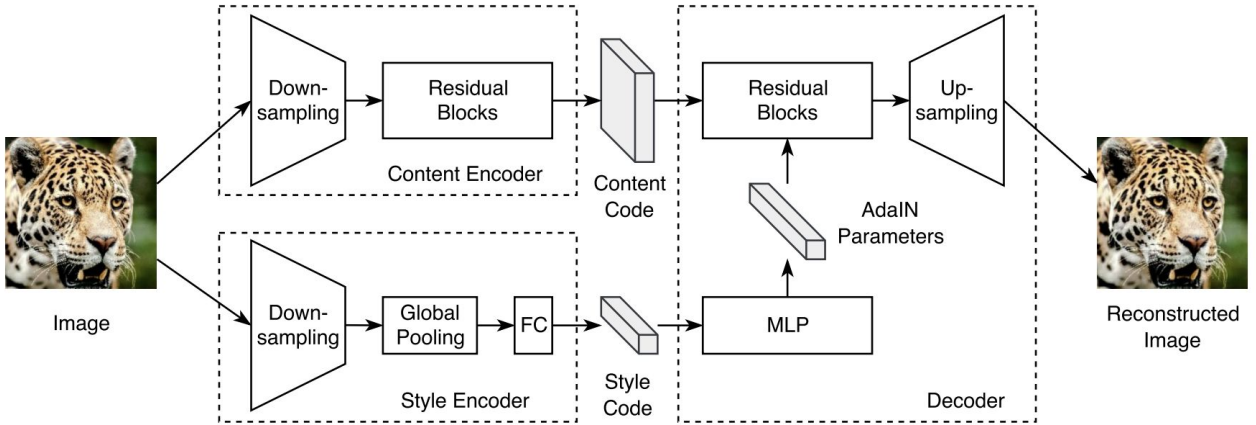
# Unsupervised Image Translation

- In unsupervised image translation we are given pairs of ($x_i$, $y_i$)
- We are not given any labels of correalted and uncorrelated attributes
- Learn latent codes to describe each type:
  - Class embedding to describe class
  - Style embedding to describe correlated attributes
  - Content embedding to describe uncorrelated attributes

$$x_i = G(e_{y_i}, s_i, c_i)$$

# Simplifying Assumption I: Structure - Appearance

- Assumption: spatial structure should be preserved across domains
- MUNIT/FUNIT: architechtural bias for content – spatial, style - global



# Loss Functions

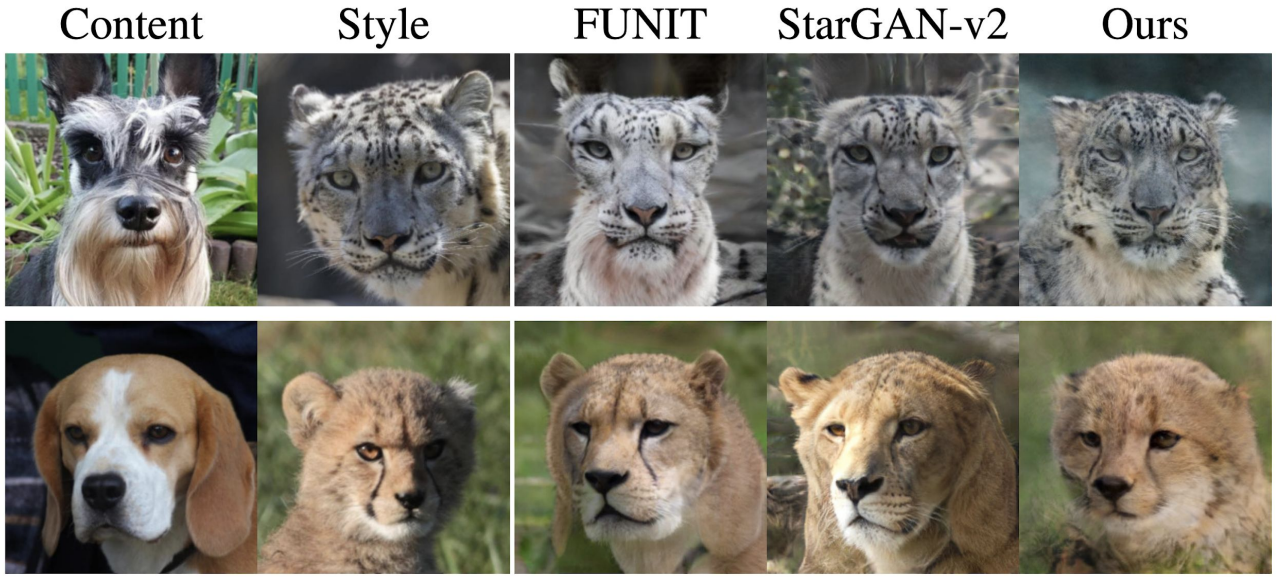
$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{x},y}\left[\log D_y(\mathbf{x})\right] + \mathbb{E}_{\mathbf{x},\widetilde{y},\mathbf{z}}[\log\left(1 - D_{\widetilde{y}}(G(\mathbf{x},\widetilde{\mathbf{s}}))\right)],$$

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{x},\widetilde{y},\mathbf{z}}\left[||\widetilde{\mathbf{s}} - E_{\widetilde{y}}(G(\mathbf{x},\widetilde{\mathbf{s}}))||_1\right]$$

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{x},y,\widetilde{y},\mathbf{z}}\left[||\mathbf{x} - G(G(\mathbf{x},\widetilde{\mathbf{s}}),\hat{\mathbf{s}})||_1\right],$$

- **Notation a bit confusing** G(x,.) in fact means G(c,.) and s=(s, y)

# Breakdown of the Spatial Architectual Bias



# Simplifying Assumption II: No Correlated Attributes

Main assumption: no uncorrelated attributes ($a_s$=0) – same as Cycle/StarGAN

All attributes apart from class can be transferred

$$x_i = G_\theta(e_{y_i}, 0, c_i)$$

**Good Assumption**

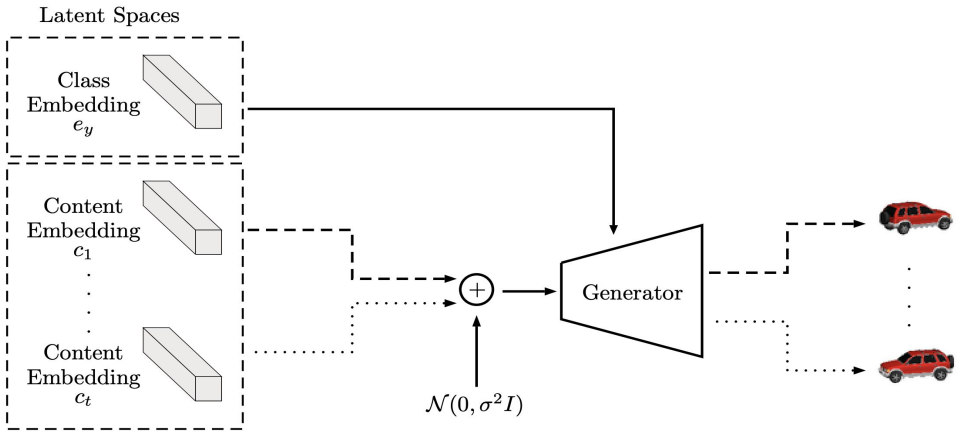

**Bad Assumption**

Gabbay and Hoshen, ICLR'20

# Shared optimization of class

Class supervision: all images $y_1,..,y_t$ of the same object, share class embedding

Class representation cannot have content info!



# Information Bottleneck on Content Representation

Shared optimization prevents content information in class code

We learn a content representation for every image, why use class at all?!

Idea: regularize information in content code

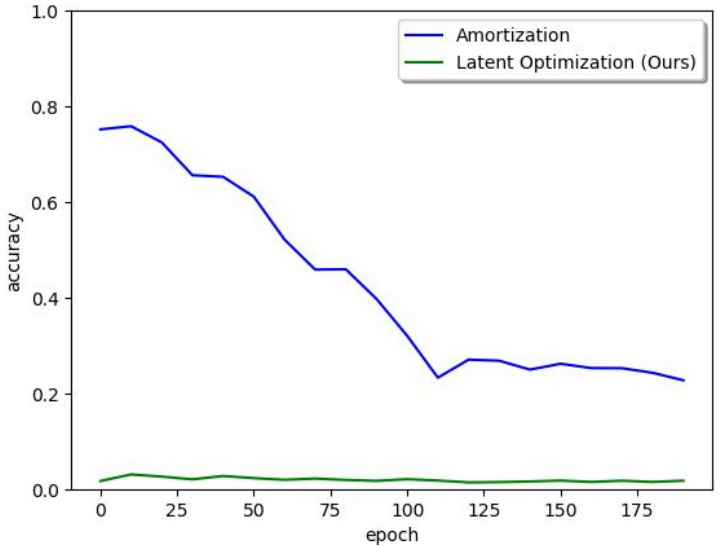
Content only contains code not possible to model in class

$$\mathcal{L} = \sum_{i=1}^{n} \|G_\theta(e_{y_i}, 0, c_i + z_i) - x_i\| + \lambda \|c_i\|^2 \qquad z_i \sim \mathcal{N}(0, \sigma^2 I)$$

**Consider difference from GLO…**

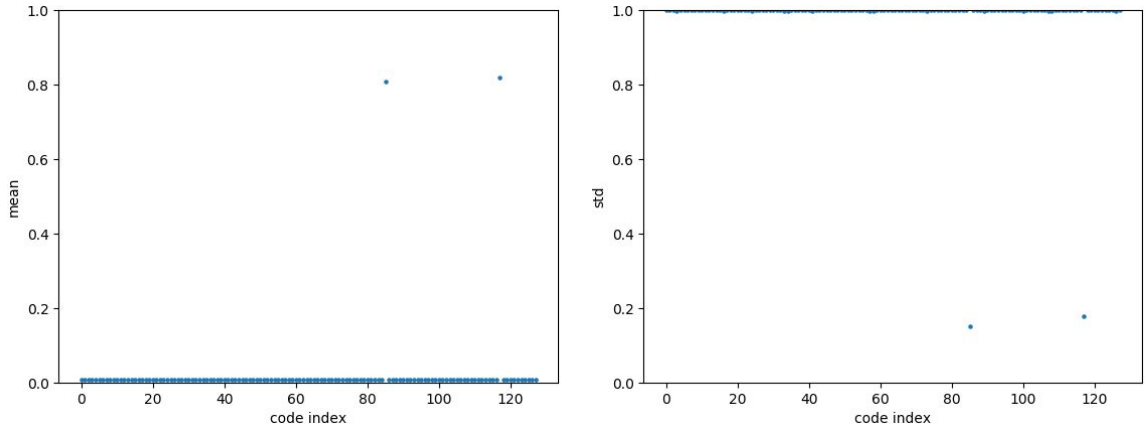
# Power of latent optimization

- A randomly initialized content encoder (for amortization) encodes class-dependent information, which needs to be minimized during training
- Random codes in latent optimization initially have zero entanglement, which is kept very low



# Fixed vs. learned variance

- Regularizing with KL-divergence often leads to partial posterior collapse:
  - Nearly all means and standard deviations learned by the encoder default to 0 and 1 respectively, carrying no information
  - A few components almost do not exhibit noise - bad for disentanglement



# KL divergence vs. Asymmetric noise

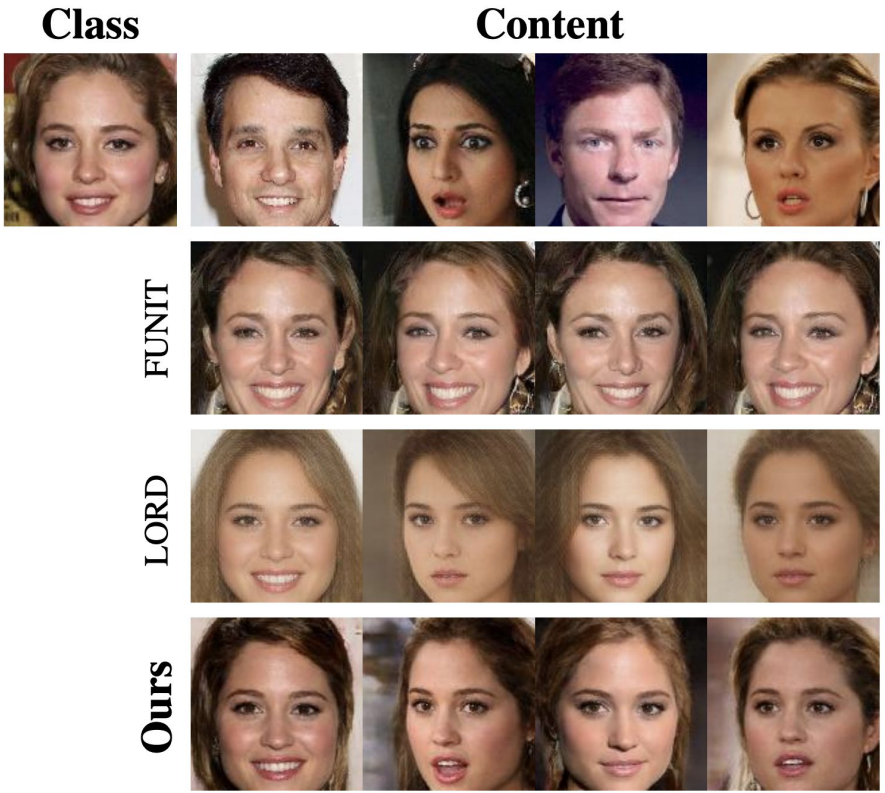
- If we assume our prior is a unit gaussian then KL-divergence terms simply translates to:

- If we set a fixed variance e.g. $\sigma = 1$ then this term translates to our activation decay penalty i.e. minimizing $\mu^2$

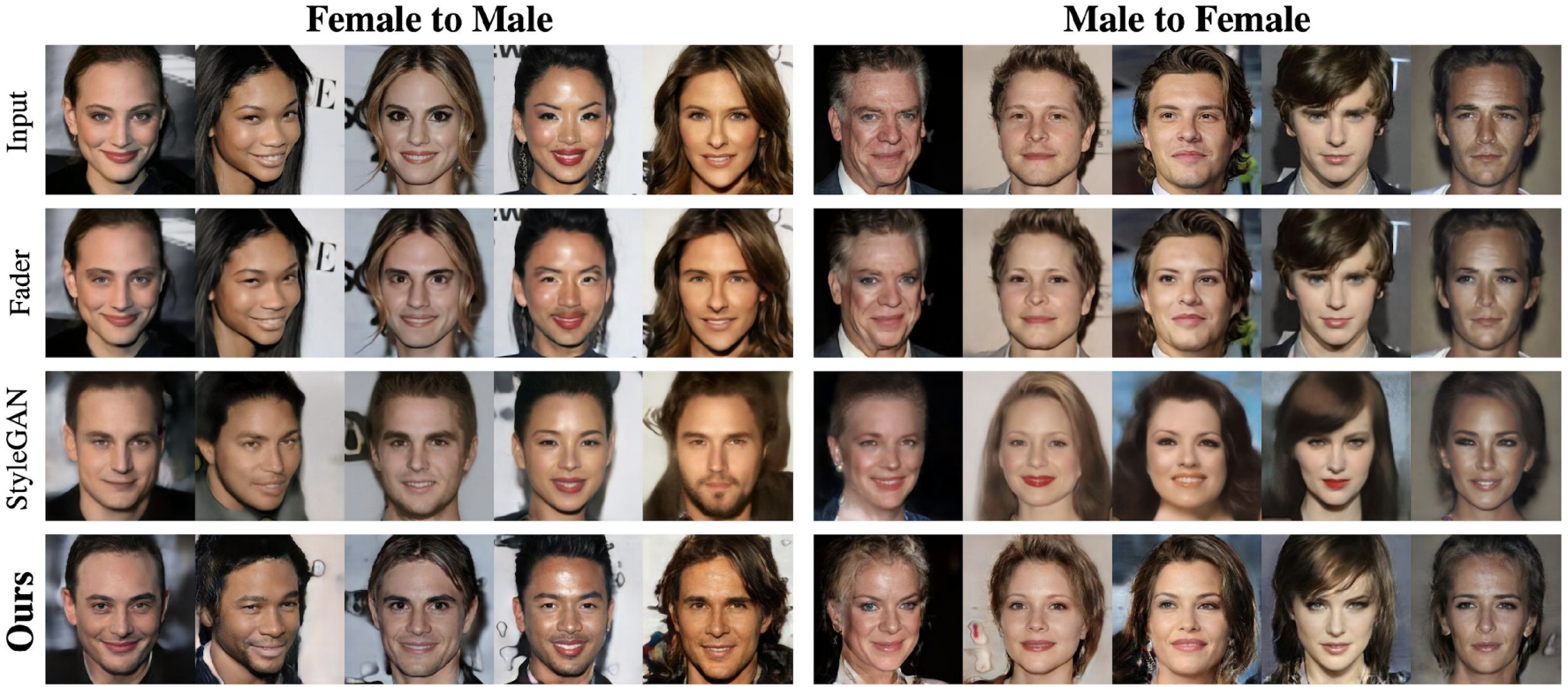
$$KL_{loss} = -\frac{1}{2}(\ln(\sigma^2) - \sigma^2 - \mu^2 + 1)$$

# Results when Assumption Holds True

- Last two lines are: without and with GAN



# Results when Assumption is So-So



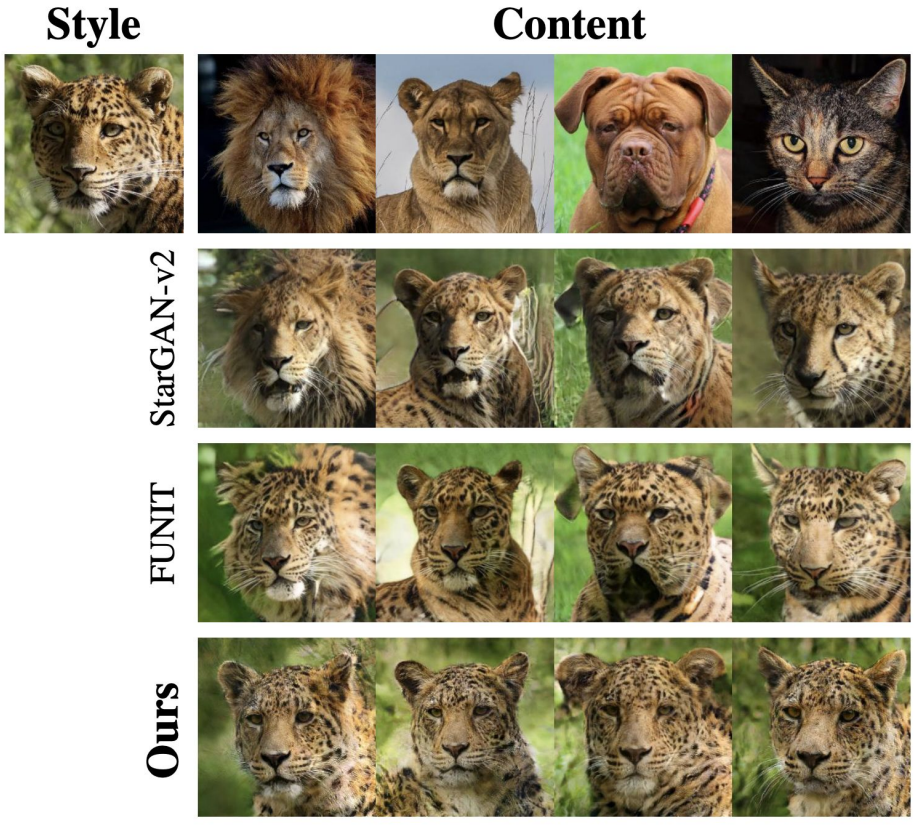
# Results when Assumption is Wrong

# Learning both Style and Content

- Assume correlated and uncorrelated attributes exist
- Instead of all uncorrelated attributes, just separate pose from all others
- Style code computed by encoding randomly transformed version of image
- Randomly cropped/flipped version of image contains no pose infromation

$$\min_{c_i', e_{y_i}, E_s, G} \mathcal{L}_{disent} = \sum_i \ell(G(e_{y_i}, E_s(x_i^{trans}), c_i' + z), x_i) + \lambda_{cb}\|c_i'\|^2 \qquad z \sim \mathcal{N}(0, I)$$

# Results: Animal Faces Transfer Works



# What’s Left to Do? Lots of open questions…

- How to translate between domains without supervision at all
    - Unsupervised disentanglement – very hard
- How to transfer some uncorrelated attributes but not others?
- How do we get rid of latent optimization?