Manipulating Natural Images by Learning Relationships between Visual Domains

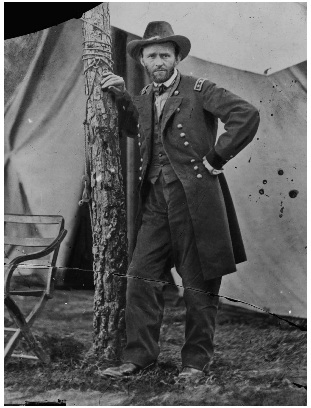
Ben Usman, Ph.D. Candidate

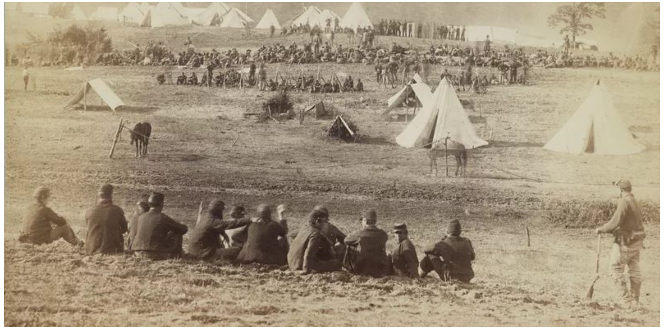
Department of Computer Science
Boston University

Cross-Domain Image Manipulation: Motivation

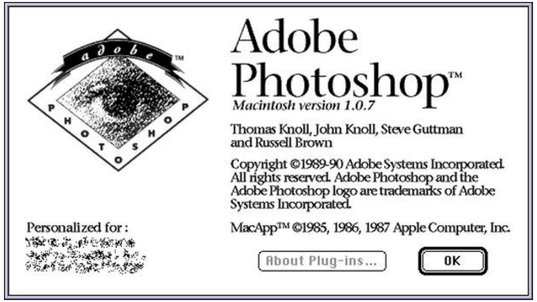
[Lenin and Trotsky during Red Square Demonstration, 1919]

[General Ulysses S. Grant, 1864]

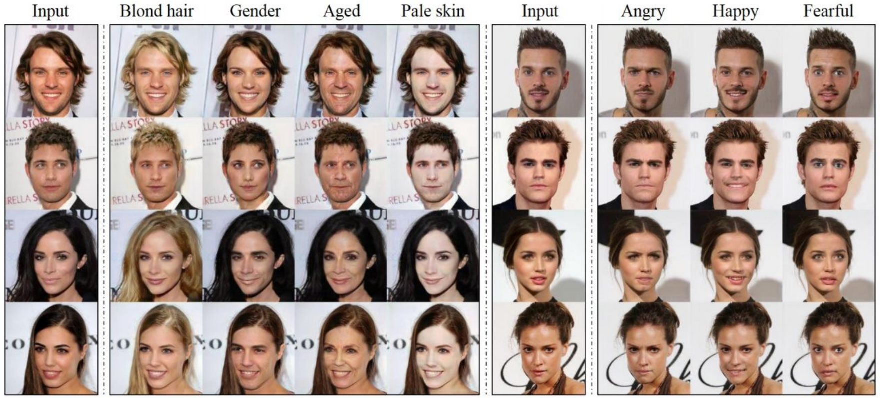




Cross-Domain Image Manipulation: Motivation

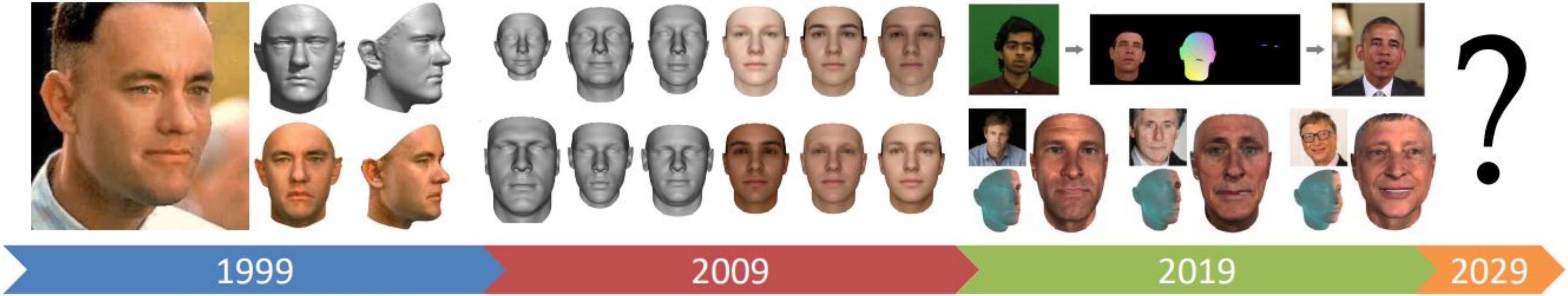


collections of **hand-crafted** manipulations



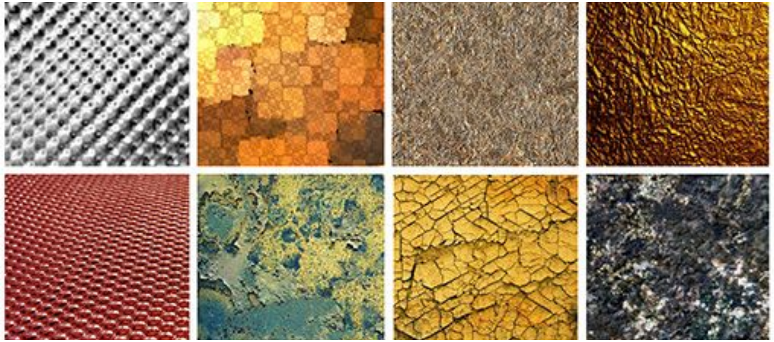
supervised neural models

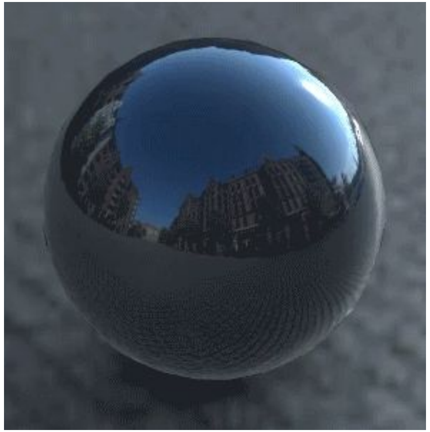
parametric models (e.g. face 3DMM)

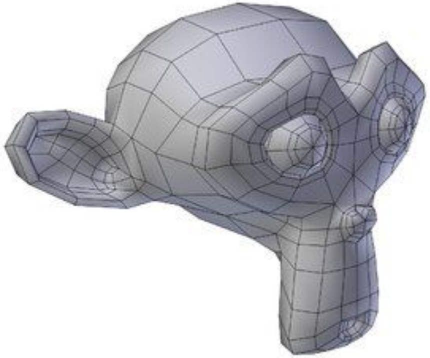


[1] “3D Morphable Face Models - Past, Present and Future”, Egger et. al, 2019
[2] “StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation”, Choi et. al, 2018

Cross-Domain Image Manipulation: Motivation





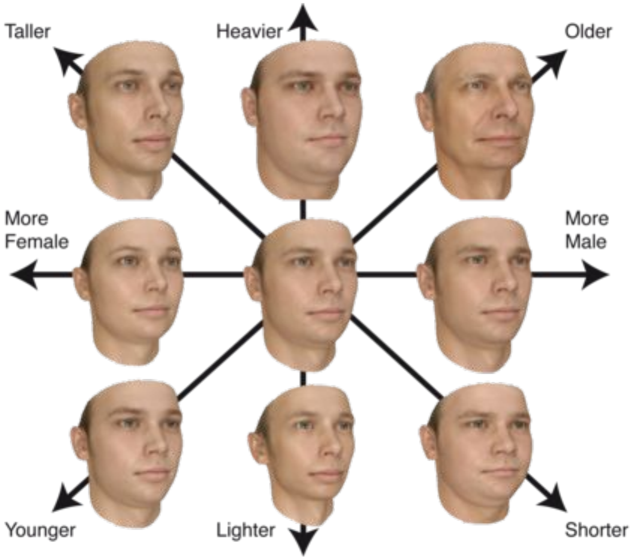


Cross-Domain Image Manipulation: Motivation

~~[0.1, 0.9, …]~~

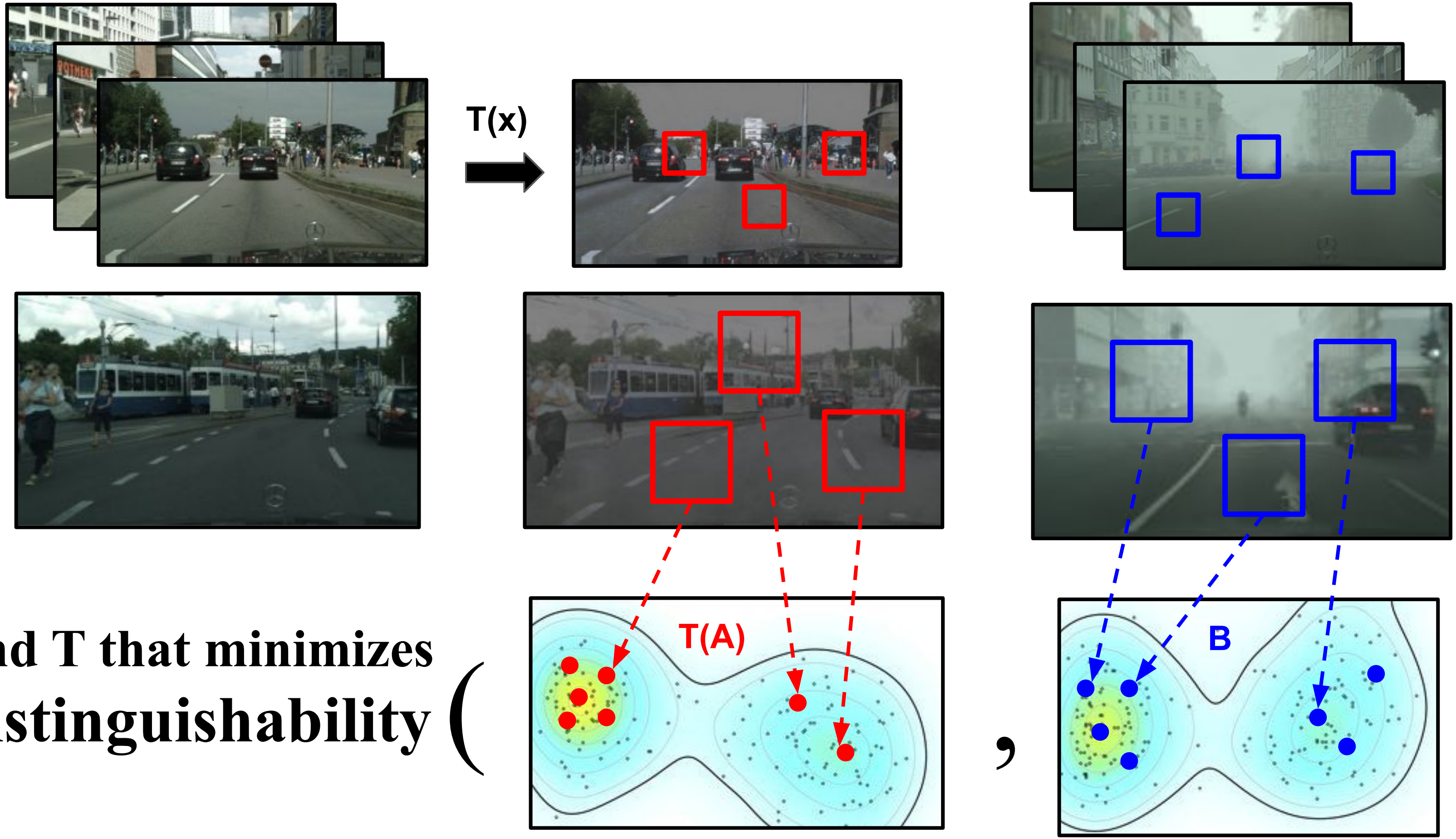
~~[0.7, 0.2, …]~~

~~[0.3, 0.1, …]~~



precise control over all attributes!

Solution: Unsupervised Image Translation / Domain Alignment



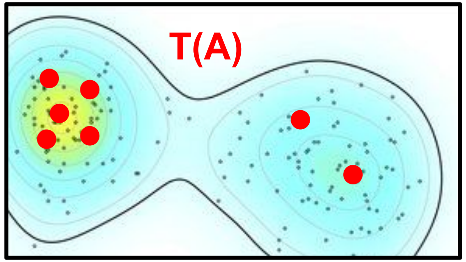
find T that minimizes distinguishability (,)

Solution: Unsupervised Image Translation / Domain Alignment

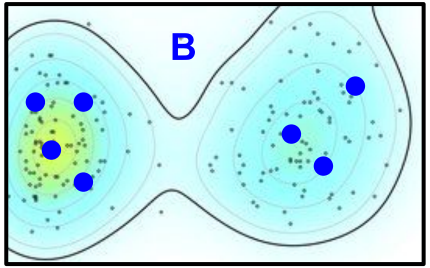
MMD, EMD, CORAL , etc.
✓ easy to optimize
✗ work poorly in higher dims

neural adversarial
✓ expressive
✗ unstable training

find T that minimizes distinguishability (

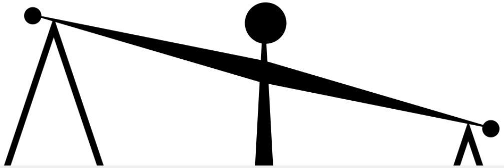


,



)

Solution: Unsupervised Image Translation / Domain Alignment



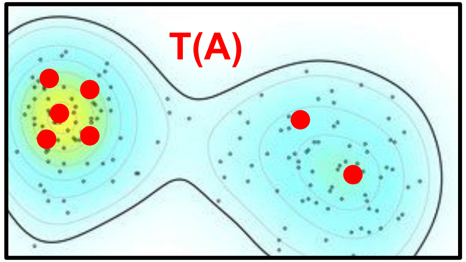
easy t

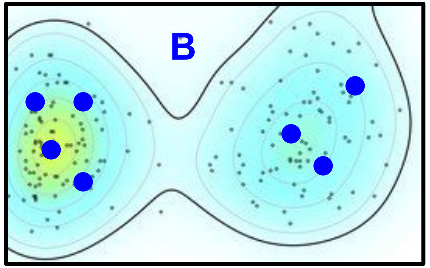
MMD, EM

e:

sarial

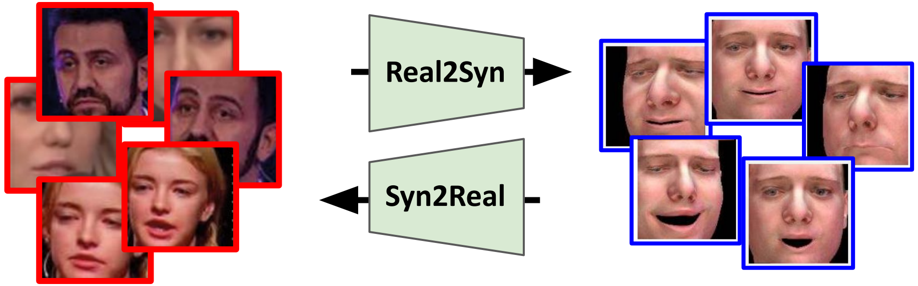
Topic 1: Stable and expressive distribution alignment.

find T that minimizes distinguishability (

, 

)

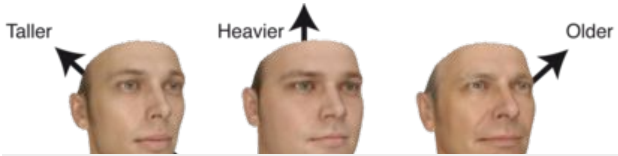
Domain Alignment: **Synthetic-to-Real**

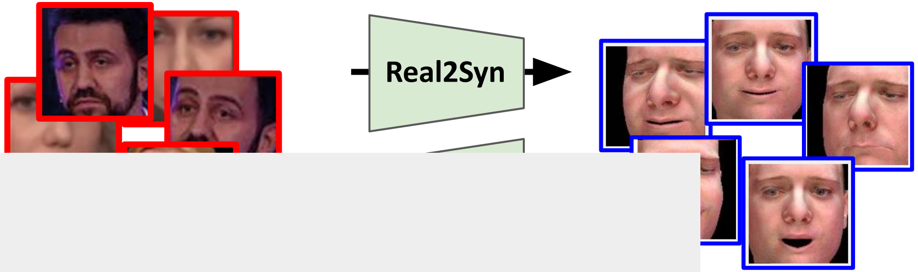


manipulate **mouth**

no precise control over attributes!

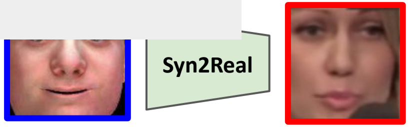
Domain Alignment: **Synthetic-to-Real**





Topic 2: Precise manipulation of real data using synthetic supervision.

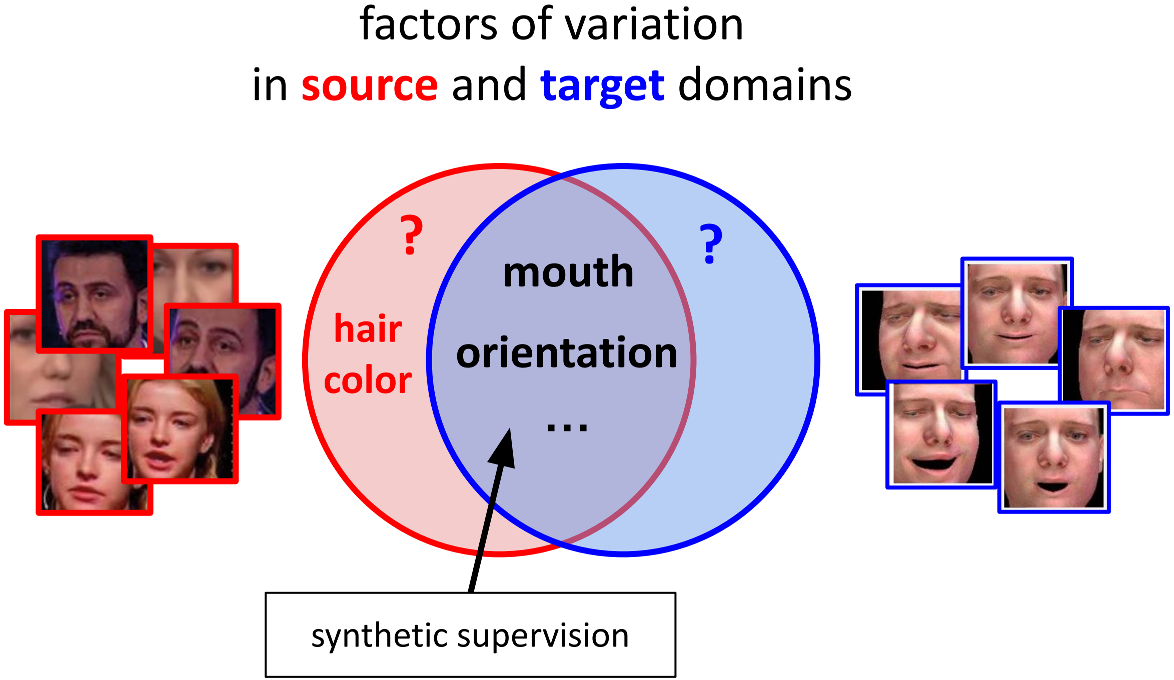
nthetic
ain



precise control over all attributes!

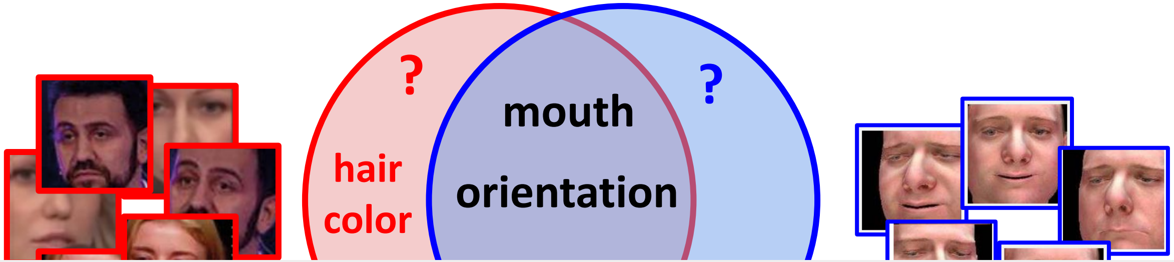
no precise control over attributes!

Domain Alignment: **Domain-specific factors**



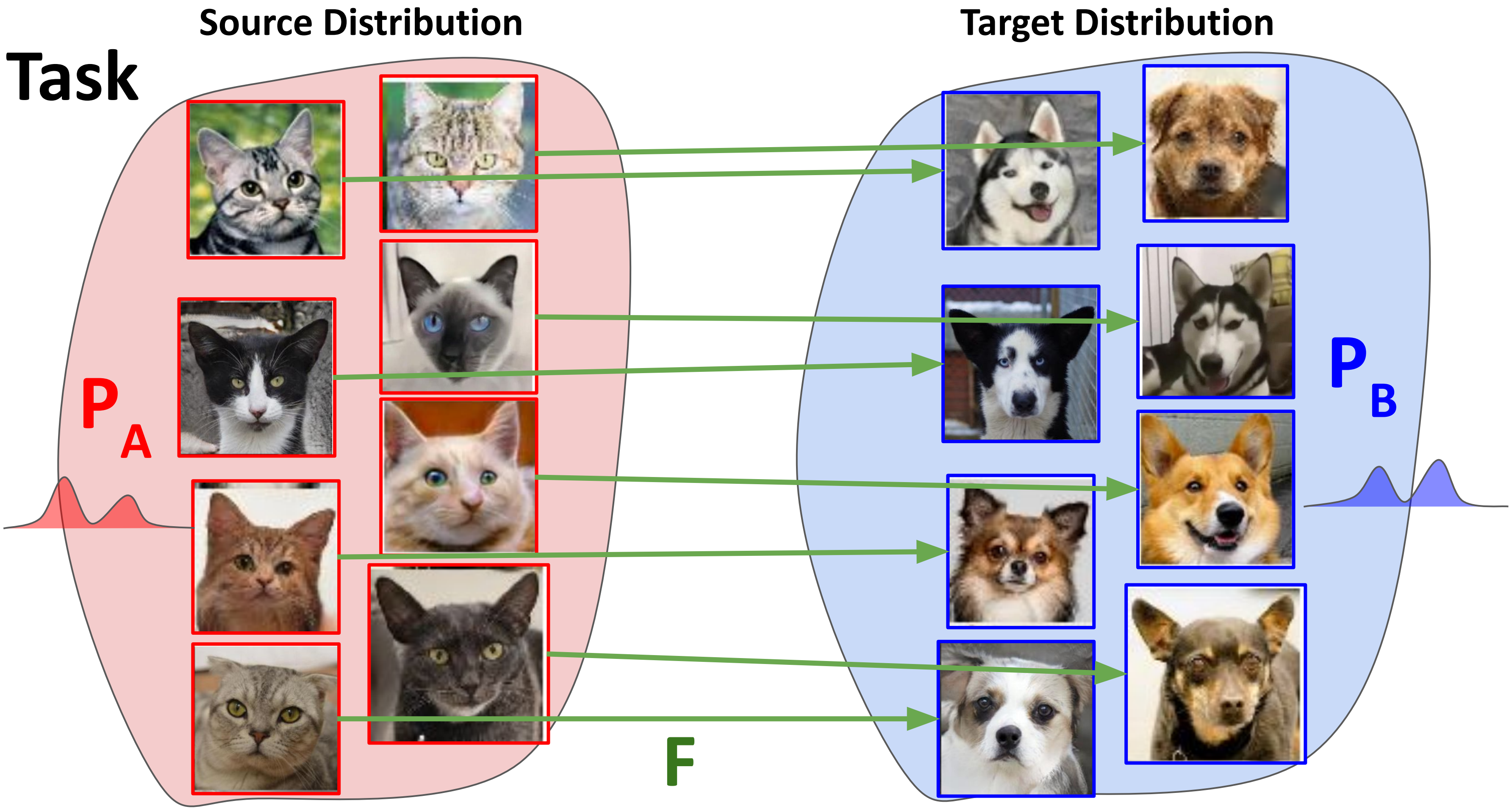
Domain Alignment: **Domain-specific factors**

factors of variation
in **source** and **target** domains



Topic 3: Disentanglement and manipulation of domain-specific and shared factors in isolation without supervision.

Task



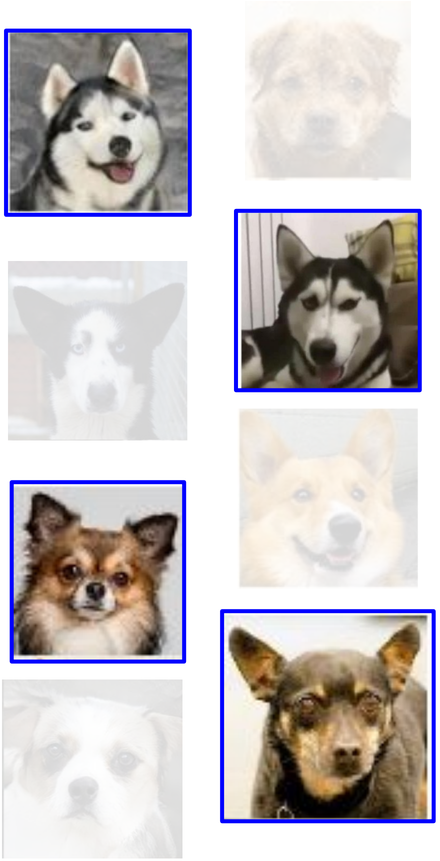
ground truth 1-to-1 cross-domain mapping

Task

Source Samples

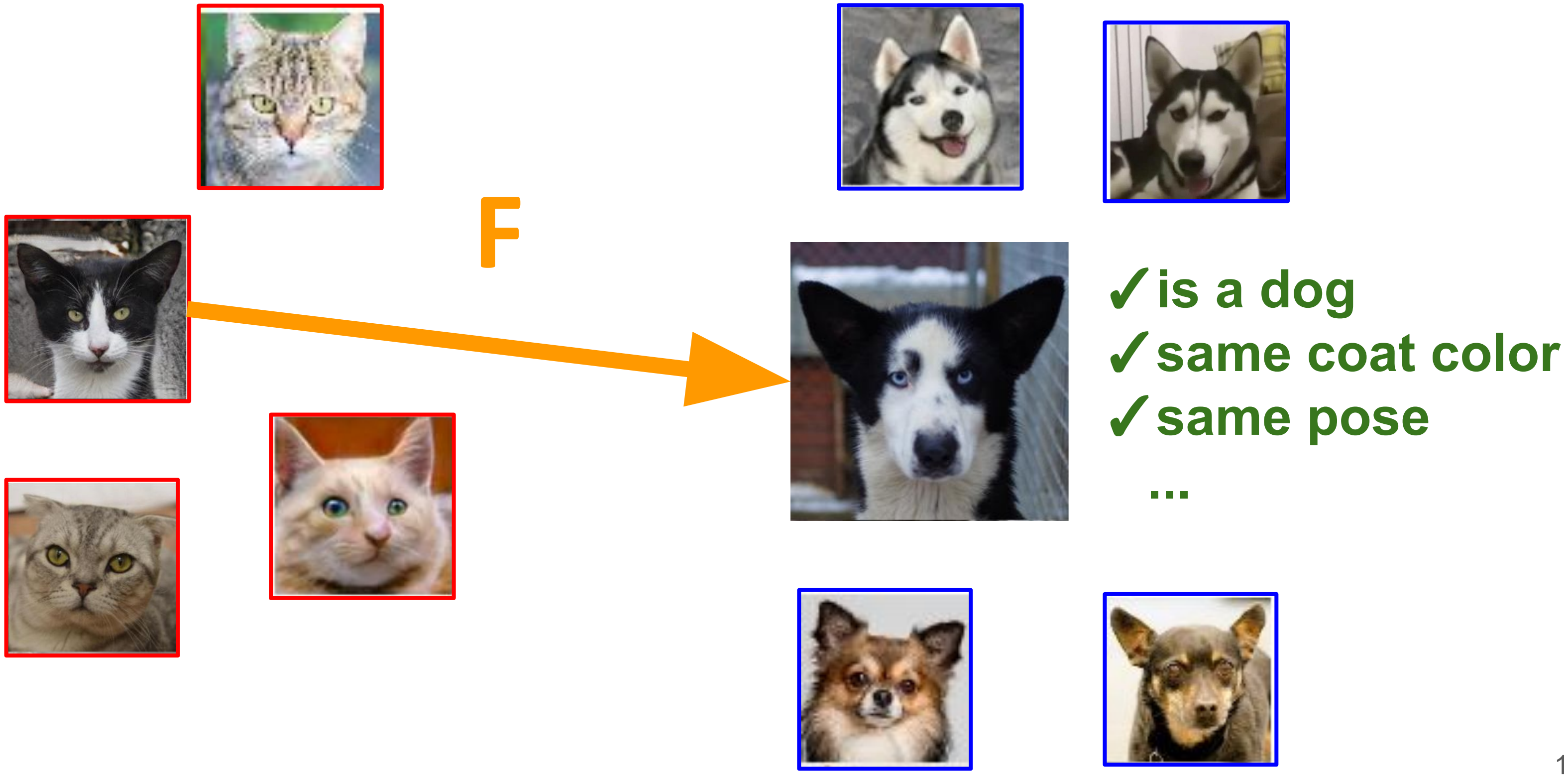
Goal: reconstruct F from unpaired samples

Target Samples



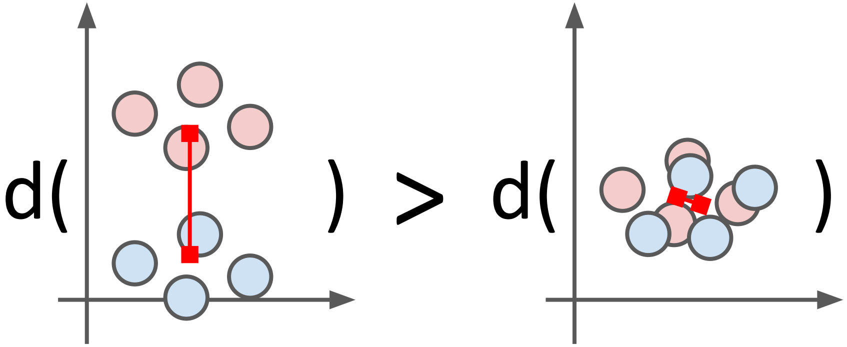
Task

Source Samples (Cats) Target Samples (Dogs)



How to find a good F?

(An empirical estimate of)
a statistical distance
d(A, B): Set, Set → ℝ

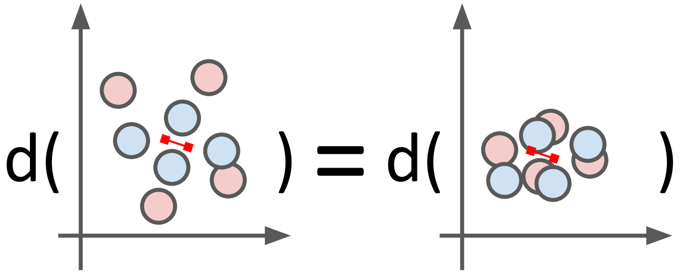


“looks like A and B are coming from different distributions”

“looks like A and B might be coming from the same distribution”

Example: difference of means

$$d(A, B) = \|\hat{\mu}(A) - \hat{\mu}(B)\|$$



“look same to me”

How to find a good F?

Source Samples

F

Translated Source Samples

minimize statistical distance d(F(A), B)

Target Samples

$A = \{a_i\}$

$F(A) = \{F(a_i) : a_i \in A\}$

$B = \{b_j\}$

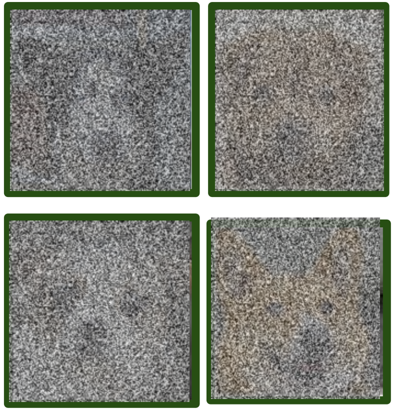
$$\min_{F \in \mathcal{F}} d(F(A), B) + R(F)$$

How to find a good F? - what we expect

Source Samples

$F_{t=0}$

Translated Source Samples



HIGH statistical distance d(F(A), B)

Target Samples

… optimizing F …

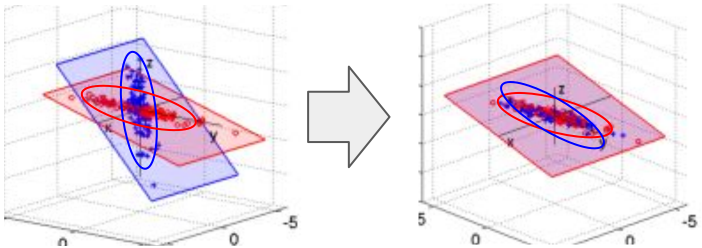
$$\min_{F \in \mathcal{F}} d(F(A), B) + R(F)$$

$F_{t=T}$

LOW statistical distance d(F(A), B)

What could go wrong?

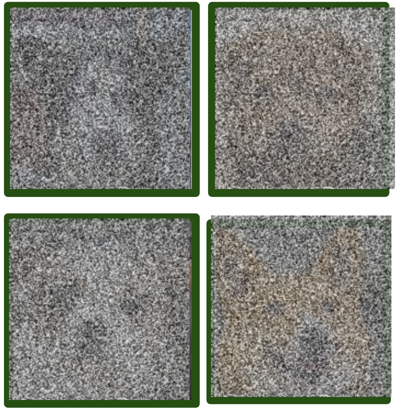
Simple parametric models are "<u>too weak</u>"



Source Samples

$F_{t=0}$

Translated Source Samples



LOW statistical distance

Target Samples

$A = \{a_i\}$

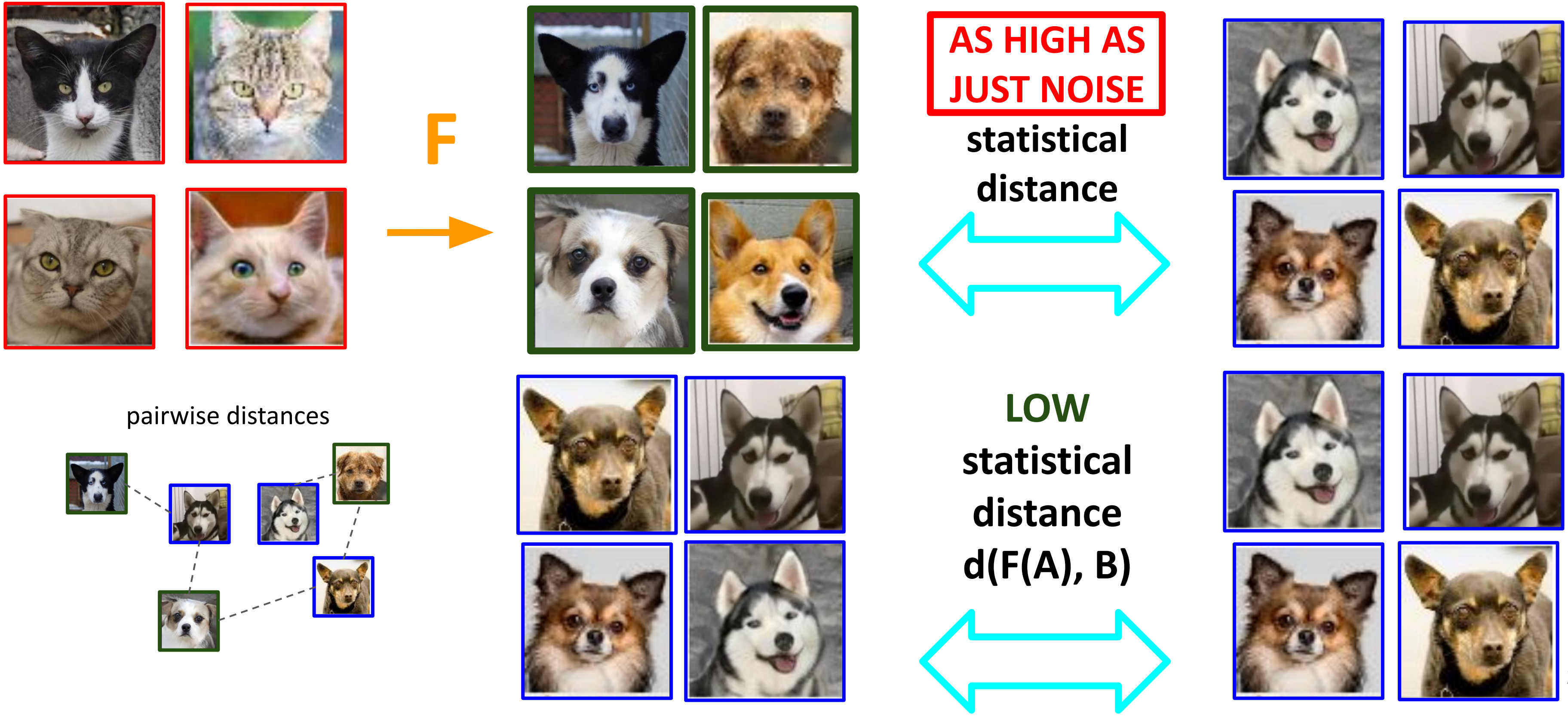
$F(A) = \{F(a_i) : a_i \in A\}$

$B = \{b_i\}$

$$\min_{F \in \mathcal{F}} d(F(A), B) + R(F)$$

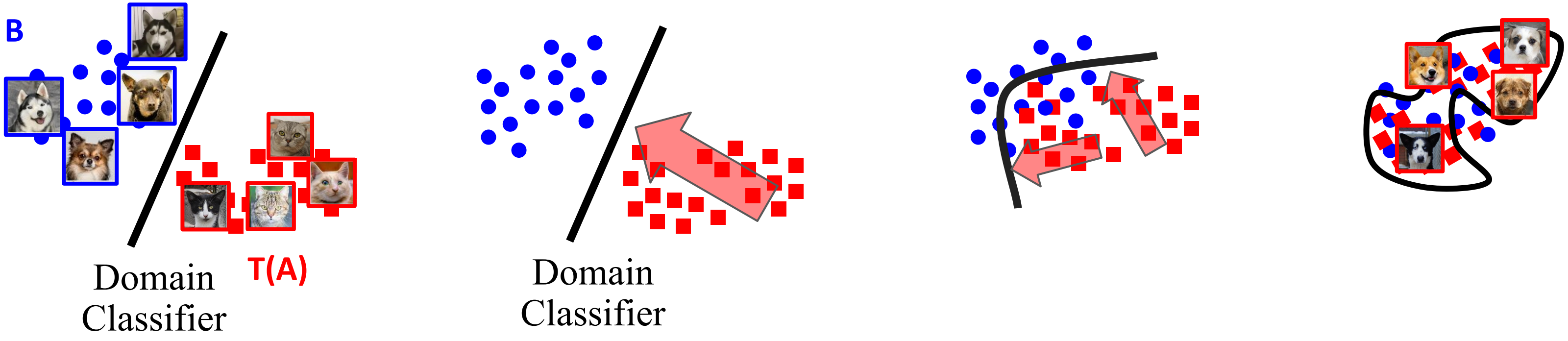
What could go wrong?

Non-parametric models (MMD, EMD) “do not generalize” well



What could go wrong?

Adversarial alignment (GANs) are unstable and fail silently



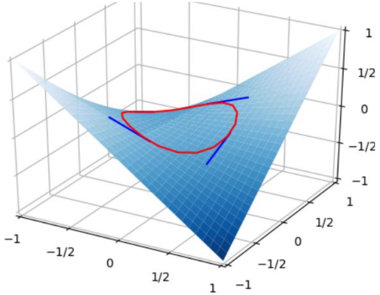
This problem is min-max!

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

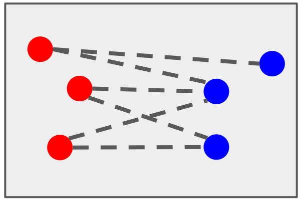
Solving min-max with 1st order methods is **hard**!

$$f(x, y) = xy$$
$$\min_x \max_y f(x, y)$$

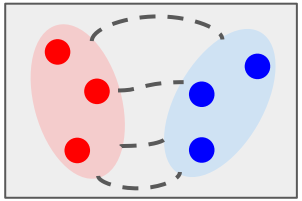
$$g(x, y) = (x, -y)$$
$$x_{t+1} = x_t + \alpha g(x, y)$$



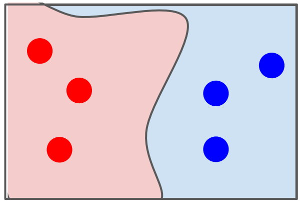
Bounding Likelihood Ratios with Normalizing Flows: Motivation



Non-parametric (MMD, EMD)



Simple Parametric (CORAL)



Adversarial (GAN, Monge–Kantorovich)

Non-parametric (MMD, EMD)	Simple Parametric (CORAL)	Adversarial (GAN, Monge–Kantorovich)
closed-form estimators exists ✓	closed-form estimators exists ✓	min-max objective
can be minimized via gradient descent ✓	can be minimized via gradient descent ✓	adversarial training
model-free (metric-based)	simple data model (e.g. normal)	powerful implicit data model (NN) ✓

Learning better one-to-one mappings

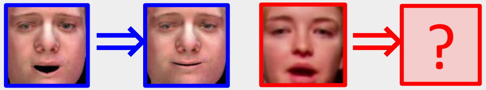
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W’18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

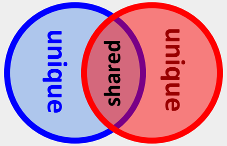
Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

Manipulating factors with cross-domain supervision

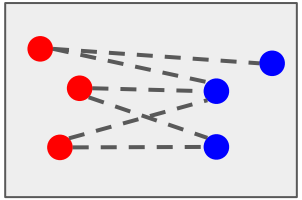
We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)



We can manipulate attributes **unique** to each domain independently from those **shared** across domains! (in submission)



Stable Alignment using Dual Adversarial Distance: Motivation

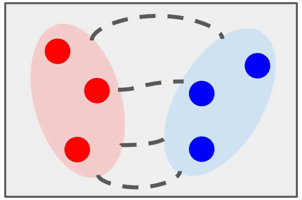


Non-parametric (MMD, EMD)

closed-form estimators exists

can be minimized via gradient descent

model-free (metric-based)

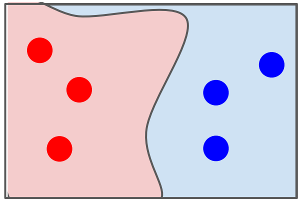


Simple Parametric (CORAL)

closed-form estimators exists

can be minimized via gradient descent

simple data model (e.g. normal)



Adversarial (GAN, Monge–Kantorovich)

min-max objective

adversarial training

powerful implicit data model (NN)

Objective Dualization

min-min objective

gradient descent

“Stable Distribution Alignment Using the Dual of the Adversarial Distance”, Usman, Saenko, Kulis (ICLR-W’18)

Stable Alignment using Dual Adversarial Distance: Our Solution

Adversarial alignment loss **for the logistic discriminator**:

$$d(A, B') = \max_w \sum_{x_i \in A} \log(\sigma(w^T x_i)) + \sum_{x_j \in B'} \log(1 - \sigma(w^T x_j)) - \frac{\lambda}{2} w^T w$$

Contribution: an equivalent dual adversarial alignment loss **for the logistic D(x).**

$$d(A, B') = \min_{0 \leq \alpha_i \leq 1/\lambda} \frac{1}{2}\alpha_A^T Q_{AA} \alpha_A + \frac{1}{2}\alpha_B^T Q_{BB} \alpha_B - \alpha_A^T Q_{AB} \alpha_B + H(\alpha_A) + H(\alpha_B)$$

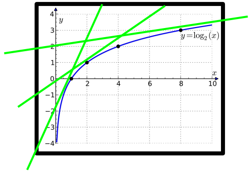
$$\text{s.t. } ||\alpha_A||_1 = ||\alpha_B||_1$$

$$Q_{AB} = A^T B \qquad H(\alpha) = \alpha^T \log \alpha + (1 - \alpha)^T \log(1 - \alpha)$$

"Stable Distribution Alignment Using the Dual of the Adversarial Distance", Usman, Saenko, Kulis (ICLR-W'18)

Stable Alignment using Dual Adversarial Distance: Our Solution

$$\max_{w,b} \sum_{x_i, y_i \in C_\theta} \log(\sigma(y_i(w^T x_i + b))) - \frac{\lambda}{2} w^T w$$



$$\boxed{\begin{aligned} &\log(\sigma(u)) \leq \alpha^T u + H(\alpha)\ , \quad \alpha_i \in [0,1] \\ &H(\alpha) = \alpha^T \log \alpha + (1-\alpha)^T \log(1-\alpha) \end{aligned}}$$

$$\min_{0 \leq \alpha \leq 1} \max_{w,b} \sum_{x_i, y_i \in C_\theta} \alpha_i y_i (w^T x_i + b) + H(\alpha) - \frac{\lambda}{2} w^T w \qquad \boxed{w^* = \tfrac{1}{\lambda}(\textstyle\sum_j x_j y_j \alpha_j)}$$

$$= \min_{0 \leq \alpha_i \leq 1} \frac{1}{2\lambda} \sum_{ij} \alpha_i \alpha_j (y_i x_i)^T (y_j x_j) + H(\alpha) = \min_{0 \leq \alpha_i \leq 1} \frac{1}{2\lambda} \alpha^T Q \alpha + H(\alpha) =$$

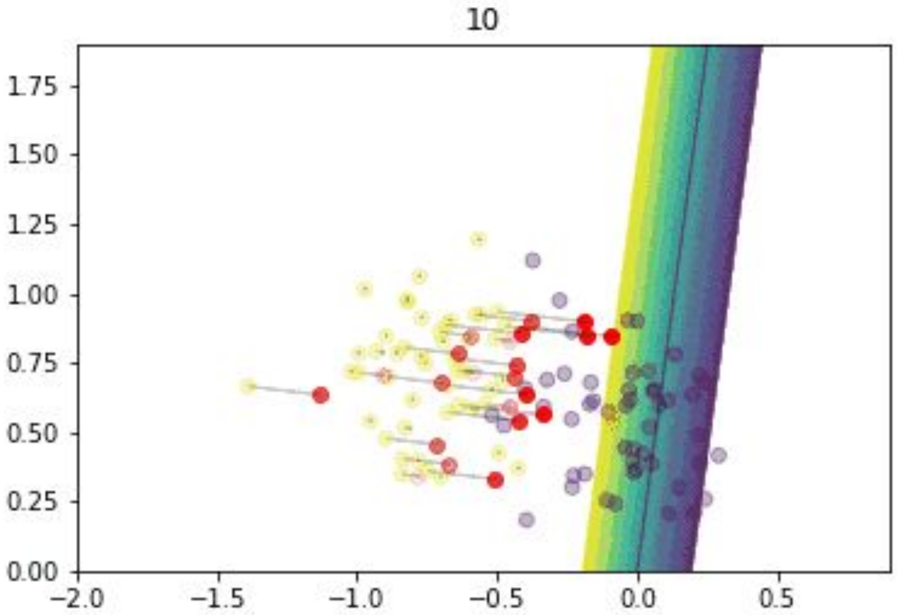
$$\min_{0 \leq \alpha_i \leq 1/\lambda} \frac{1}{2} \alpha_A^T Q_{AA} \alpha_A + \frac{1}{2} \alpha_B^T Q_{BB} \alpha_B - \alpha_A^T Q_{AB} \alpha_B + H(\alpha_A) + H(\alpha_B)$$

$$\text{s.t. } ||\alpha_A||_1 = ||\alpha_B||_1$$

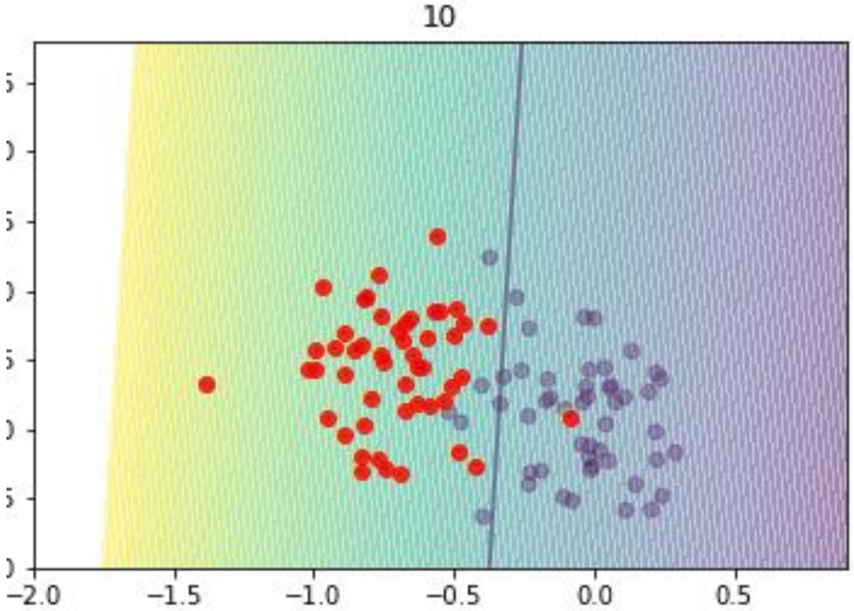
“Stable Distribution Alignment Using the Dual of the Adversarial Distance”, <u>Usman</u>, Saenko, Kulis (ICLR-W’18)

Stable Alignment using Dual Adversarial Distance: Experiments

Linear dual



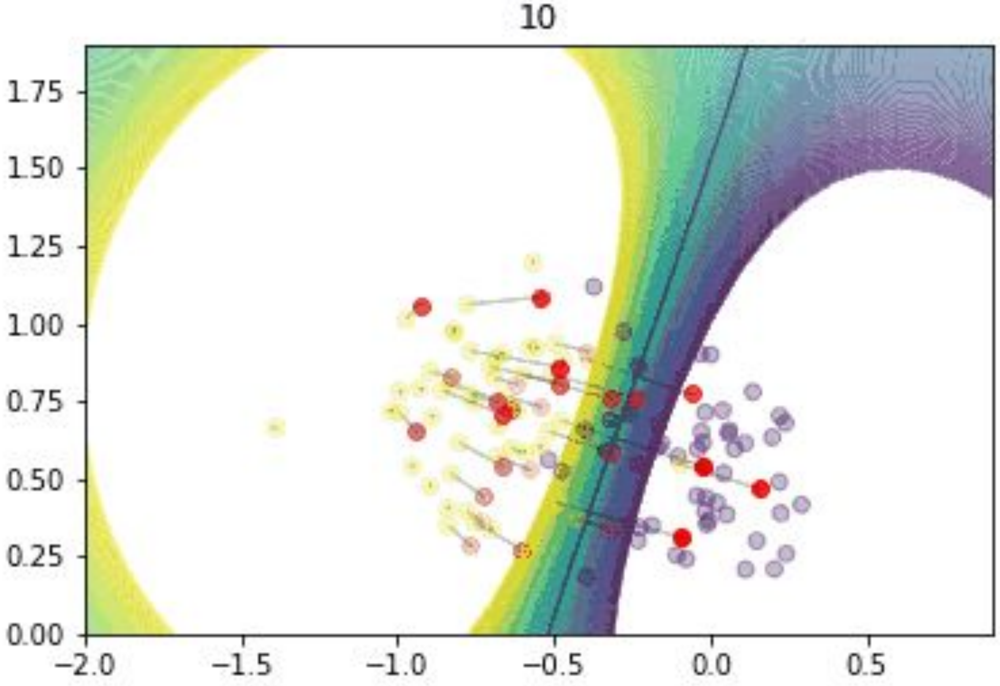
Linear min-max



"Stable Distribution Alignment Using the Dual of the Adversarial Distance", Usman, Saenko, Kulis (ICLR-W'18)

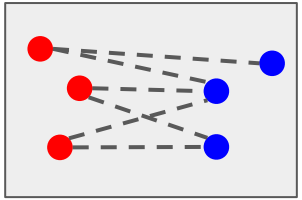
Stable Alignment using Dual Adversarial Distance: Experiments

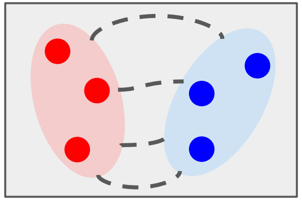
Kernel dual

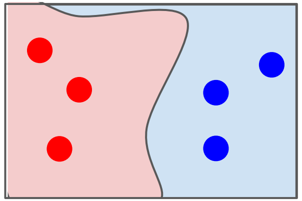


"Stable Distribution Alignment Using the Dual of the Adversarial Distance", Usman, Saenko, Kulis (ICLR-W'18)

Stable Alignment using Dual Adversarial Distance: **Takeaway**







Non-parametric (MMD, EMD)	Simple Parametric (CORAL)	Adversarial (GAN, Monge–Kantorovich)	Dual Adversarial Distance
closed-form estimators exists	closed-form estimators exists	min-max objective	min-min objective
can be minimized via gradient descent	can be minimized via gradient descent	adversarial training	gradient descent
model-free (metric-based)	simple data model (e.g. normal)	powerful implicit data model (NN)	again, relatively simple data model

Learning better one-to-one mappings

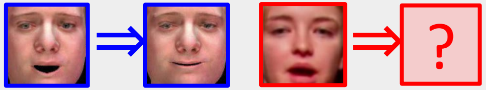
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W’18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

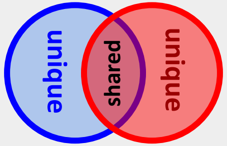
Manipulating factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)

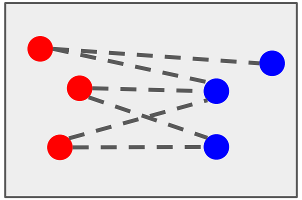
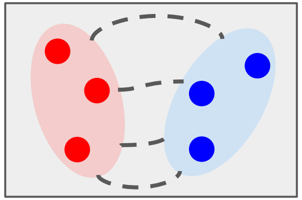
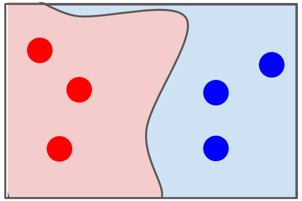
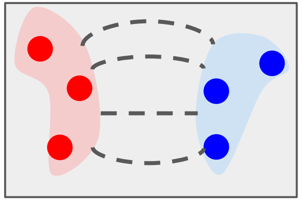


We can manipulate attributes **unique** to each domain independently from those **shared** across domains!

(in submission)

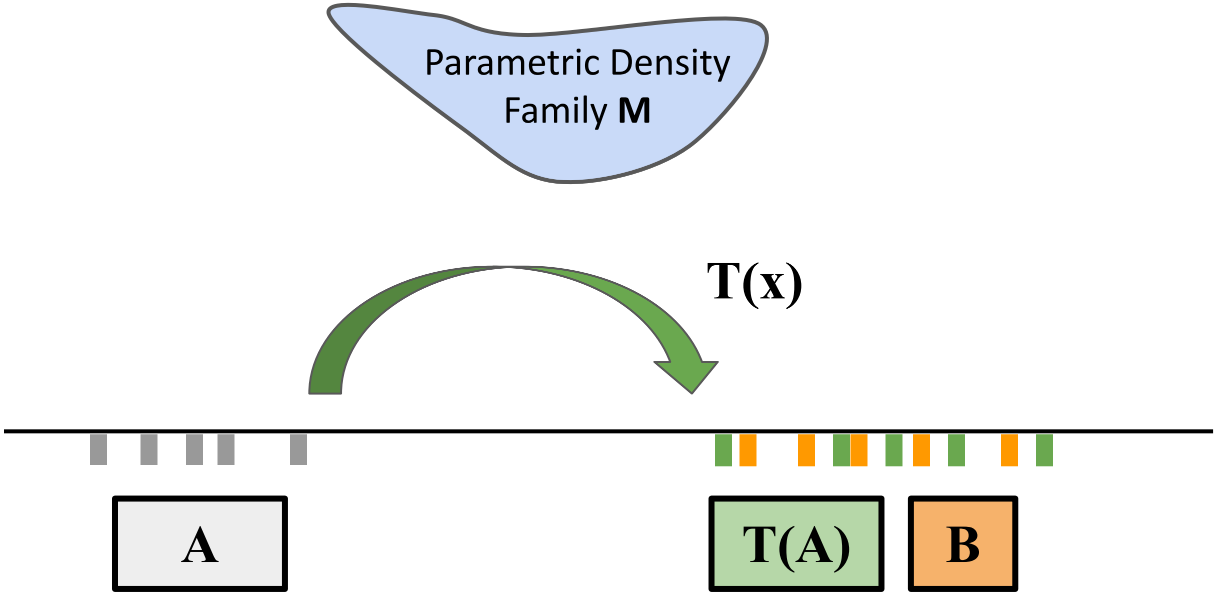


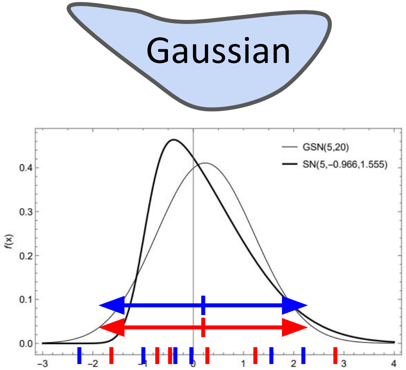
Bounding Likelihood Ratios with Normalizing Flows: Motivation

			
Non-parametric (MMD, EMD)	**Simple Parametric (CORAL)**	**Adversarial (GAN, Monge–Kantorovich)**	**Log-Likelihood Ratio Minimizing Flows (new!)**
closed-form estimators exists ✓	closed-form estimators exists ✓	min-max objective	closed-form upper bound ✓
can be minimized via gradient descent ✓	can be minimized via gradient descent ✓	adversarial training	can be minimized via gradient descent ✓
model-free (metric-based)	simple data model (e.g. normal)	powerful implicit data model (NN) ✓	any tractable density + normalizing flow ✓

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: Method





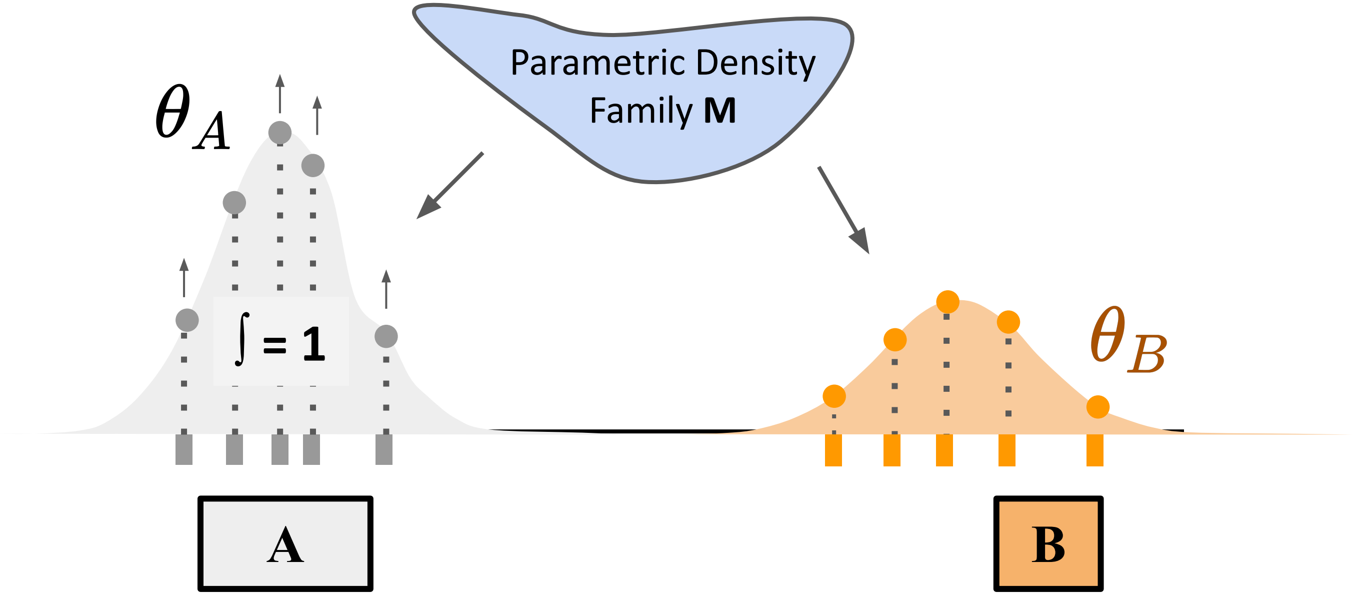
"look same to me"

Goal:

find T such that **T(A)** and **B** are **indistinguishable for M**.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

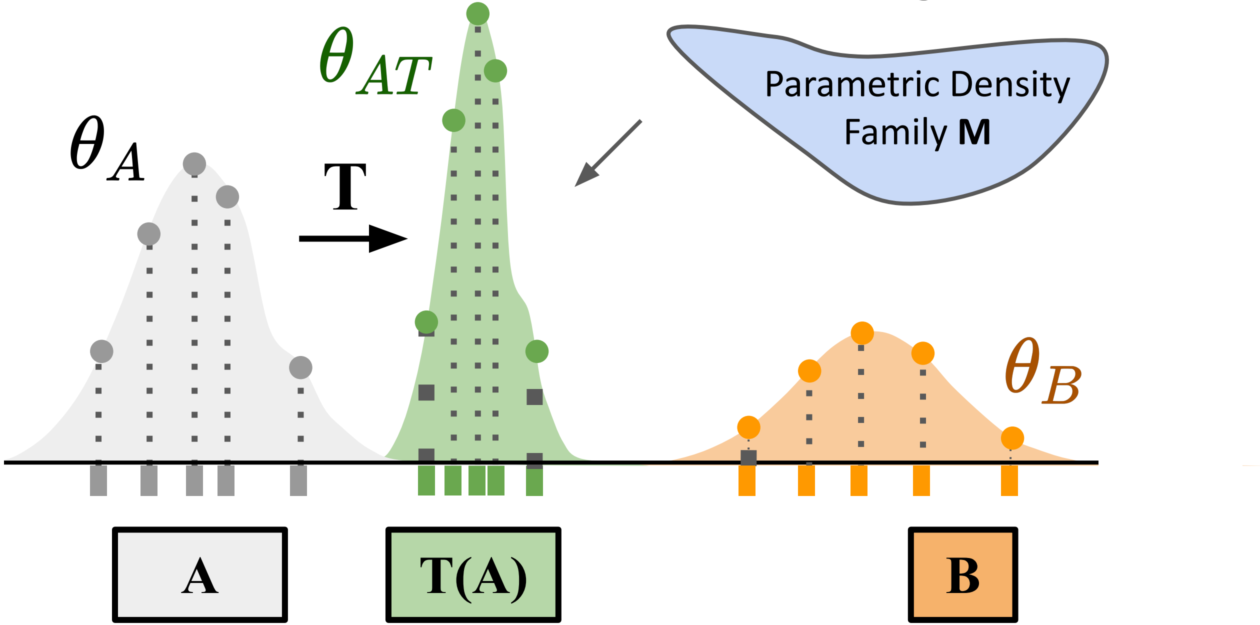
Bounding Likelihood Ratios with Normalizing Flows: Method



We start with two dataset **A** and **B** that we want to align.
We assume that we fitted two separate density models with parameters $\boldsymbol{\theta}_A$ and $\boldsymbol{\theta}_B$ to each dataset individually.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

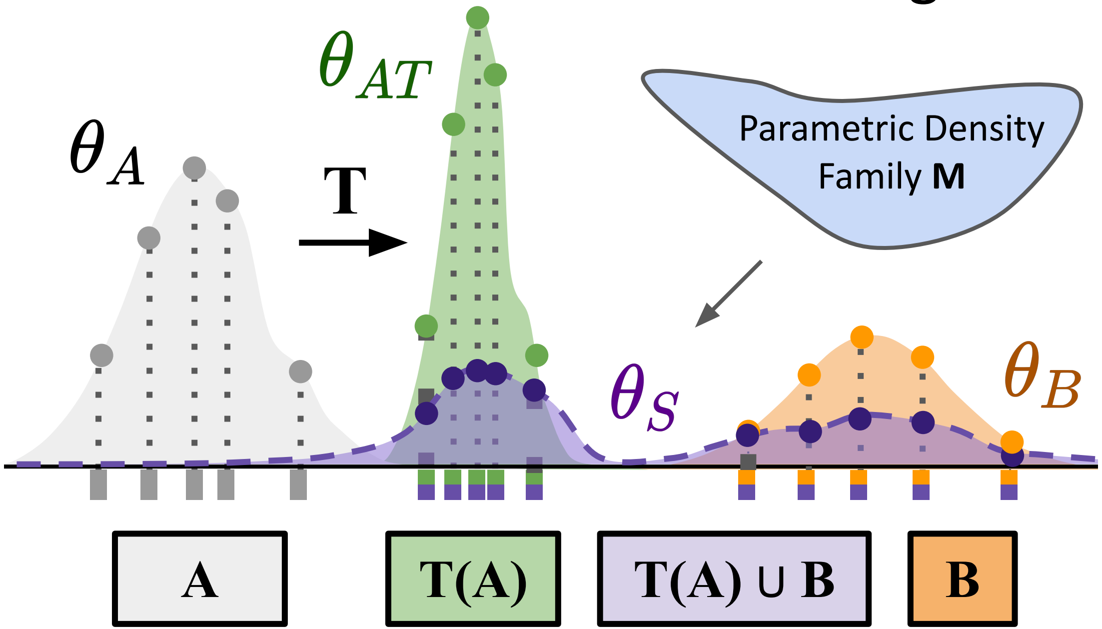
Bounding Likelihood Ratios with Normalizing Flows: Method



Then we introduce the "transformed" distribution **T(A)**
and fit a density model to it.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

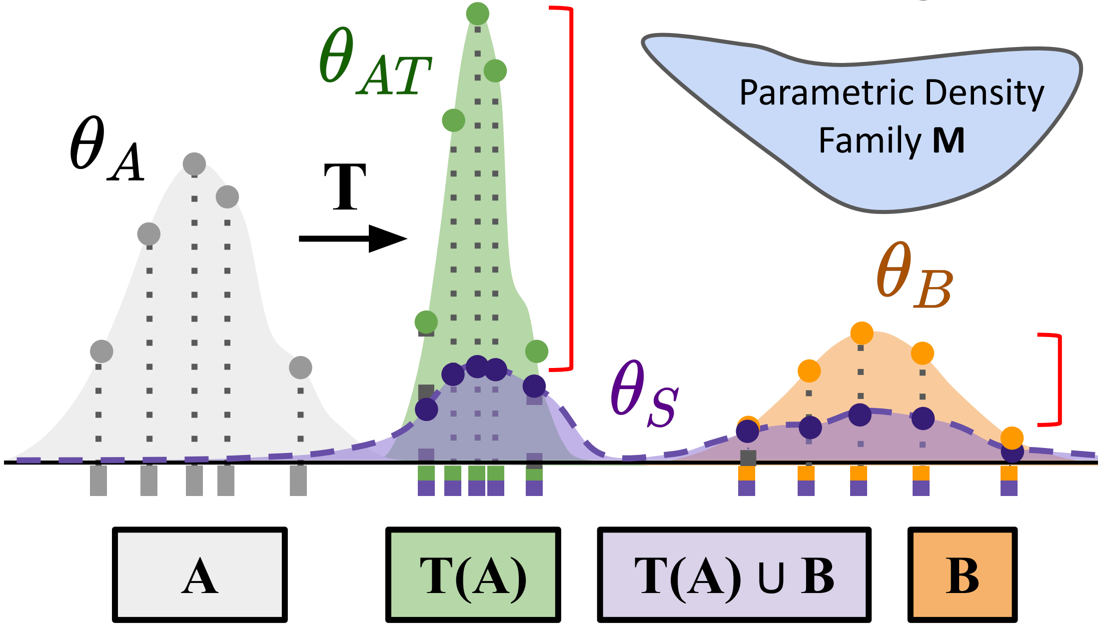
Bounding Likelihood Ratios with Normalizing Flows: Method



Then we introduce the "shared" density model **S** fit to the "combined" dataset **T(A)** ∪ **B**.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: Method

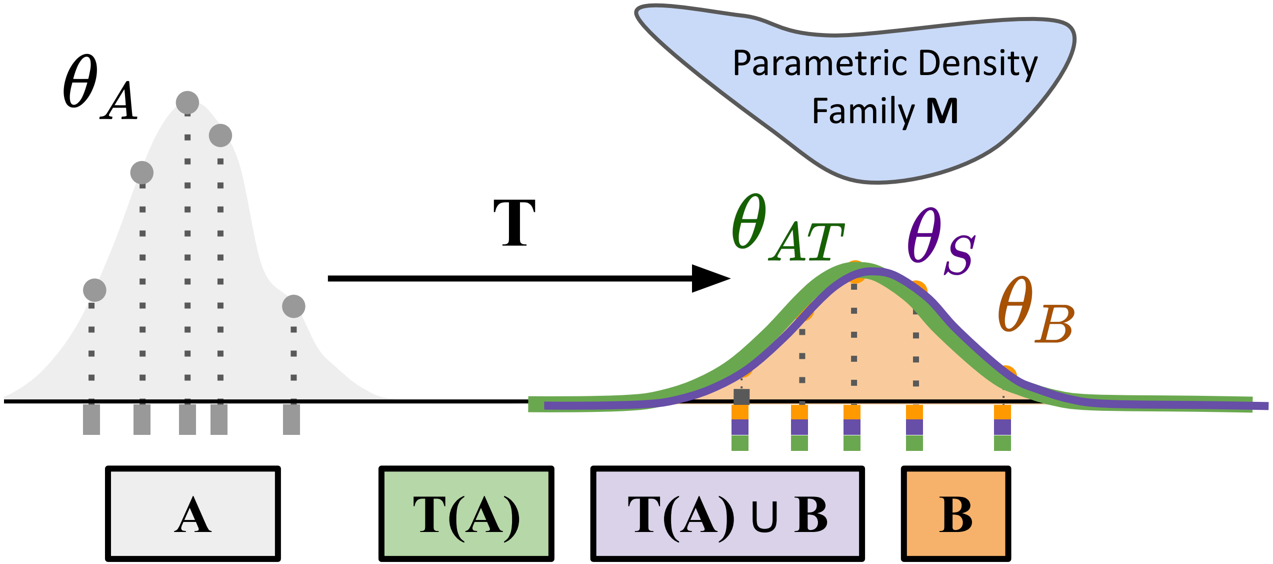


Observation 1 (⇒ Lemma 2.1):

The likelihood of the "shared" model (**S**) trained on the "combined" dataset is always **lower** than likelihoods of "private" models trained on each dataset alone (**T(A)**, **B**), **unless** both datasets are from the **same** distribution.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: Method

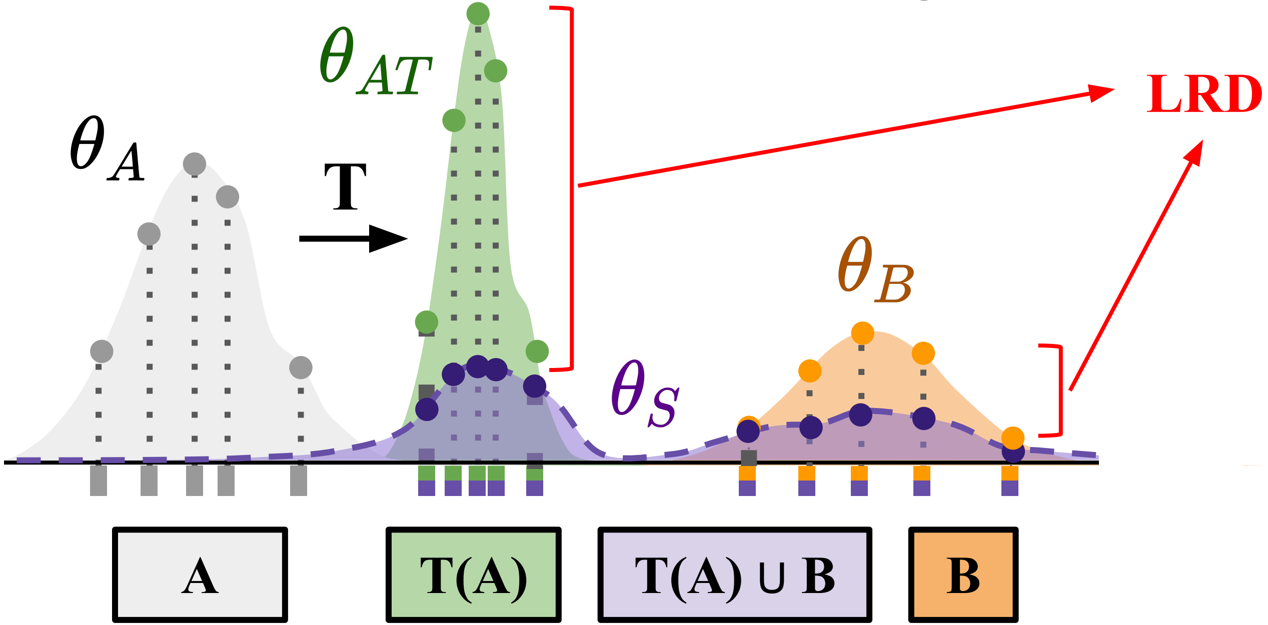


Observation 1 (⇒ Lemma 2.1):

The likelihood of the "shared" model (**S**) trained on the "combined" dataset is always **lower** than likelihoods of "private" models trained on each dataset alone (**T(A)**, **B**), **unless** both datasets are from the **same** distribution.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

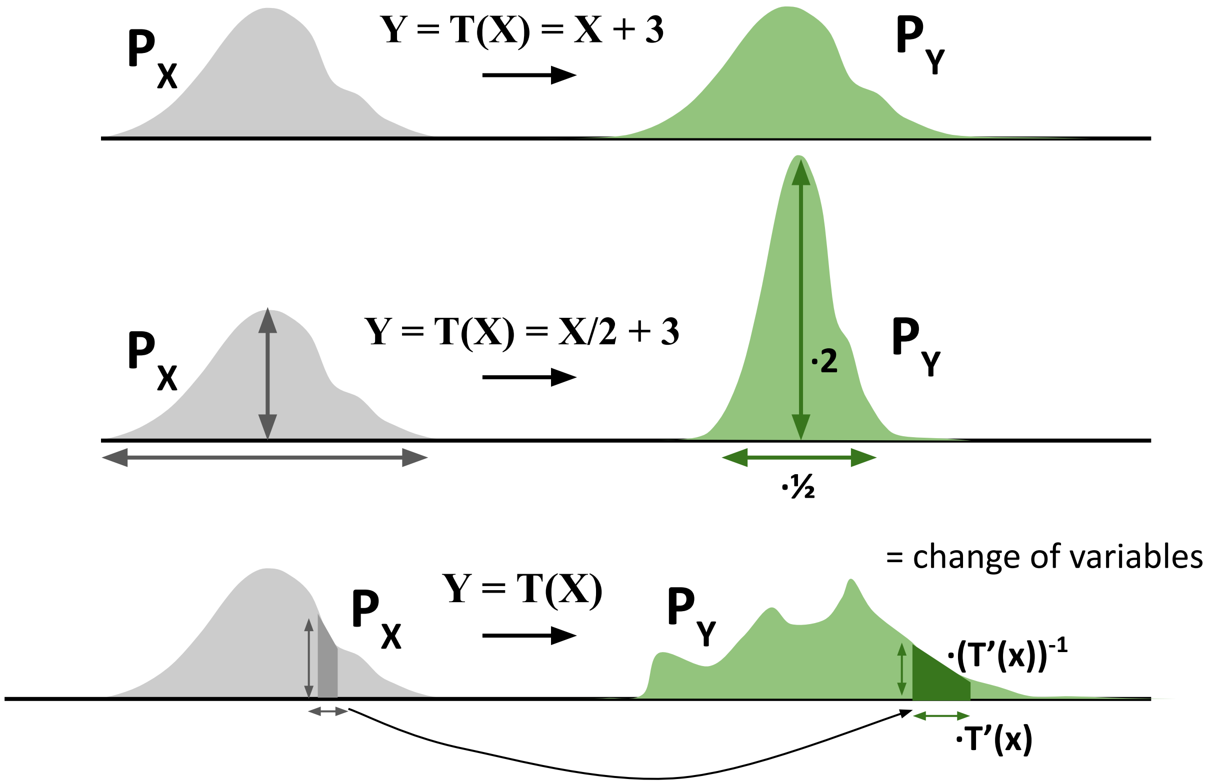
Bounding Likelihood Ratios with Normalizing Flows: Method



Definition 1 (Likelihood-Ratio Distance):
LR-distance between **T(A)** and **B** equals the difference between log-likelihoods of the optimal "shared" density **S** fit to the combined **T(A)** ∪ **B** and two optimal "private" densities fit to **T(A)** and **B** independently.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: Background

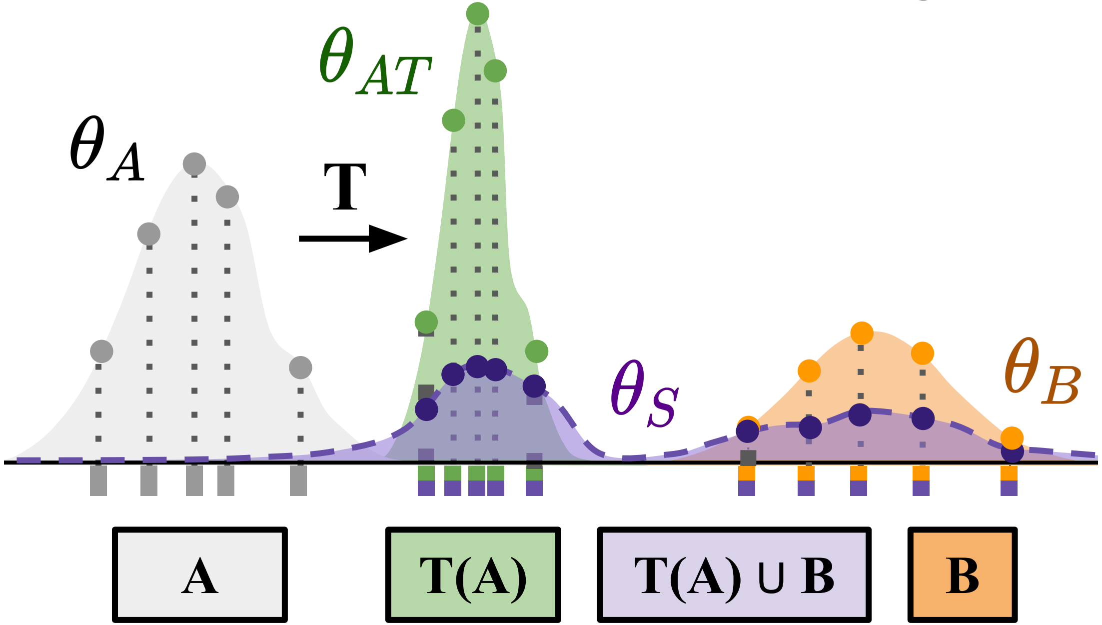


Normalizing flows:
- efficiently invertible
- efficient computation of det Jac[T](x)

⇒ can apply change of variables formula efficiently!

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: Method

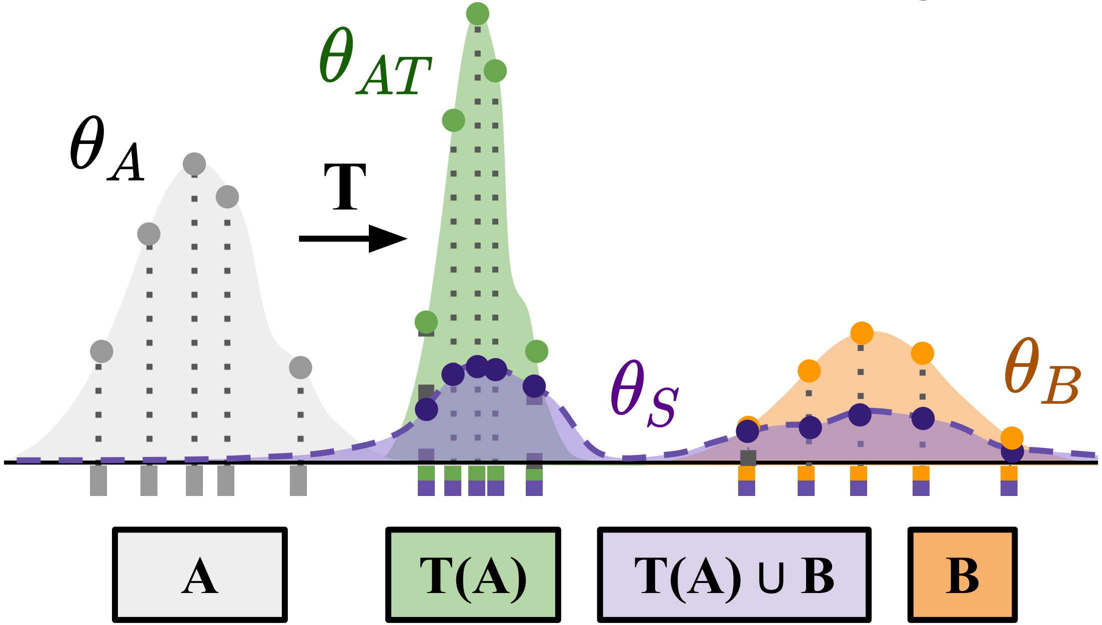


Observation 2 (⇒ Lemma 2.2):

The optimal likelihood of the transformed dataset T(A) can be approximated
in closed-form if T(x) is a **normalizing flow**.

“Log-Likelihood Ratio Minimizing Flows”, <u>Usman</u>, Sud, Dufour, Saenko (NeurIPS’20)

Bounding Likelihood Ratios with Normalizing Flows: Method



Conclusion (⇒ Theorem 2.3):

We can find the optimal flow T* that minimizes the adversarial LR-distance (the "gap" between shared private likelihoods) by minimizing a adversarial LRMF

$$\mathcal{L}_{\mathrm{LRMF}}(A, B, \phi, \theta_S) = -\log\det|\nabla_x T(A;\phi)| - \log P_M(T(A;\phi);\theta_S) - \log P_M(B;\theta_S) + c(A, B)$$

Bounding Likelihood Ratios with Normalizing Flows: Method

Lemma 2.2. *If $T(x;\phi)$ is a normalizing flow, then the first term in the objective (1) can be bounded in closed form as a function of ϕ up to an approximation error $\mathcal{E}_{bias}$. The equality in (2) holds when the approximation term vanishes, i.e. if M approximates both A and $T(A;\phi)$ equally well; P_A is the true distribution of A and $T[P_A,\phi]$ is the push-forward distribution of the transformed dataset.*

$$\max_{\theta_{AT}} \log P_M(T(A;\phi);\theta_{AT}) \le \max_{\theta_A} \log P_M(A;\theta_A) - \log\det|\nabla_x T(A;\phi)| \boxed{+\, \mathcal{E}_{bias}(A,T,M) \quad (2)}$$

$$\mathcal{E}_{bias}(A,T,M) \triangleq \max_{\phi}\left[\min_{\theta} \mathcal{D}_{KL}(P_A; M(\theta)) - \min_{\theta} \mathcal{D}_{KL}(T[P_A,\phi]; M(\theta))\right]$$

“Log-Likelihood Ratio Minimizing Flows”, Usman, Sud, Dufour, Saenko (NeurIPS’20)

Bounding Likelihood Ratios with Normalizing Flows: Method

First, we add and remove the true (unknown) entropy $H[P_A] = -\mathbb{E}_{a\sim P_A} \log P_A(a)$:

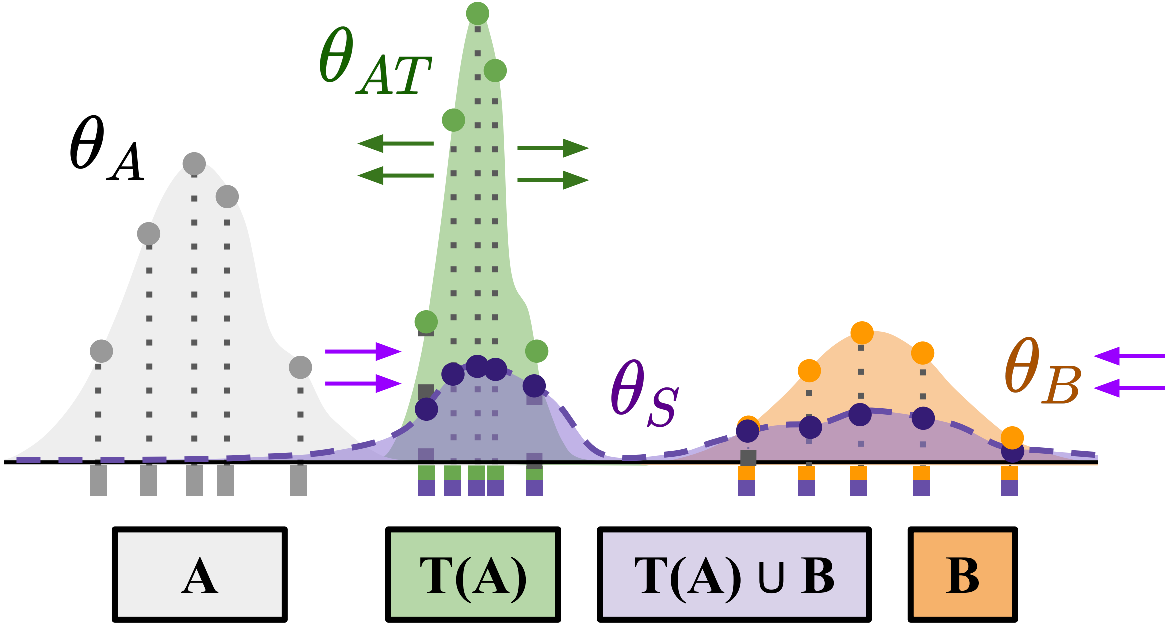
$$\max_{\theta_A} \mathbb{E}_{a\sim P_A} \log P_M(a;\theta_A) = \max_{\theta_A} \left[\mathbb{E}_{a\sim P_A} \log P_A(a) - \mathbb{E}_{a\sim P_A} \log \frac{P_A(a)}{P_M(a;\theta_A)}\right]$$
$$= H[P_A] - \min_{\theta_A} \mathbb{E}_{a\sim P_A} \left[\log \frac{P_A(a)}{P_M(a;\theta_A)}\right] = H[P_A] - \min_{\theta} \mathcal{D}_{KL}(P_A; M(\theta)). \qquad (\star)$$

And then add and remove the (unknown) entropy of the transformed distribution $H[T[P_A,\phi]]$. We also use the change of variable formula $\boxed{T[P_A](x) = P_A(T^{-1}(x)) \cdot \det |\nabla_x T^{-1}(x)|}$, and substitute the expression for $H[P_A]$ from the previous line $(\star)$:

$$\max_{\theta_{AT}} \log P_M(T(A;\phi);\theta_{AT}) = \max_{\theta_{AT}} \mathbb{E}_{a'\sim T[P_A,\phi]} \log P_M(a';\theta_{AT})$$
$$= \max_{\theta_{AT}} \left[\mathbb{E}_{a'\sim T[P_A,\phi]} \log T[P_A](a') - \mathbb{E}_{a'\sim T[P_A,\phi]} \log \frac{T[P_A,\phi](a')}{P_M(a';\theta_{AT})}\right]$$
$$= \max_{\theta_{AT}} \Big[\mathbb{E}_{a\sim P_A} P_A(T^{-1}(T(a,\phi),\phi)) +$$
$$+ \log\det |\nabla_x T^{-1}(T(a,\phi),\phi)| - \mathcal{D}_{KL}(T[P_A,\phi]; M(\theta_{AT}))\Big]$$
$$= H[P_A] - \log\det |\nabla_x T(A,\phi)| - \min_{\theta} \mathcal{D}_{KL}(T[P_A,\phi]; M(\theta))$$
$$\leq \max_{\theta_A} \log P_M(A;\theta_A) - \log\det |\nabla_x T(A,\phi)| + \mathcal{E}_{bias}(A,T,M).$$

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

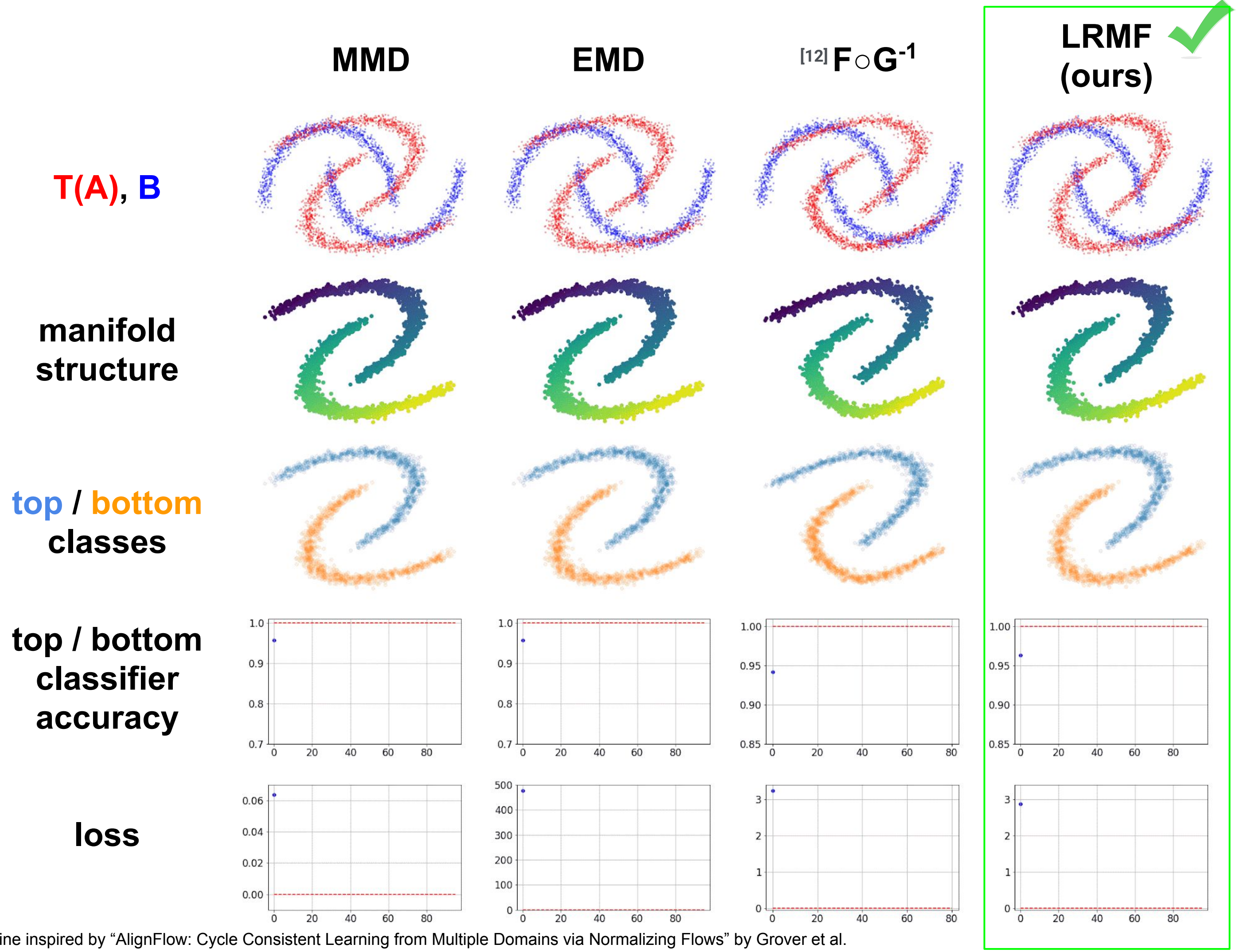
Bounding Likelihood Ratios with Normalizing Flows: Method



Conclusion (⇒ Theorem 2.3):

We can find the optimal flow T* that minimizes the adversarial LR-distance (the "gap" between shared private likelihoods) by minimizing a

$$\mathcal{L}_{\mathrm{LRMF}}(A, B, \phi, \theta_S) = -\log\det|\nabla_x T(A;\phi)| - \log P_M(T(A;\phi);\theta_S) - \log P_M(B;\theta_S) + c(A, B)$$



[12] baseline inspired by “AlignFlow: Cycle Consistent Learning from Multiple Domains via Normalizing Flows” by Grover et al.

Bounding Likelihood Ratios with Normalizing Flows: Special Cases

Special cases:

1. Gaussian LRMF $\Leftrightarrow$ matching mean and variance.

2. Minimizing an infinite capacity LRMF loss $\Leftrightarrow$ minimizing JSD

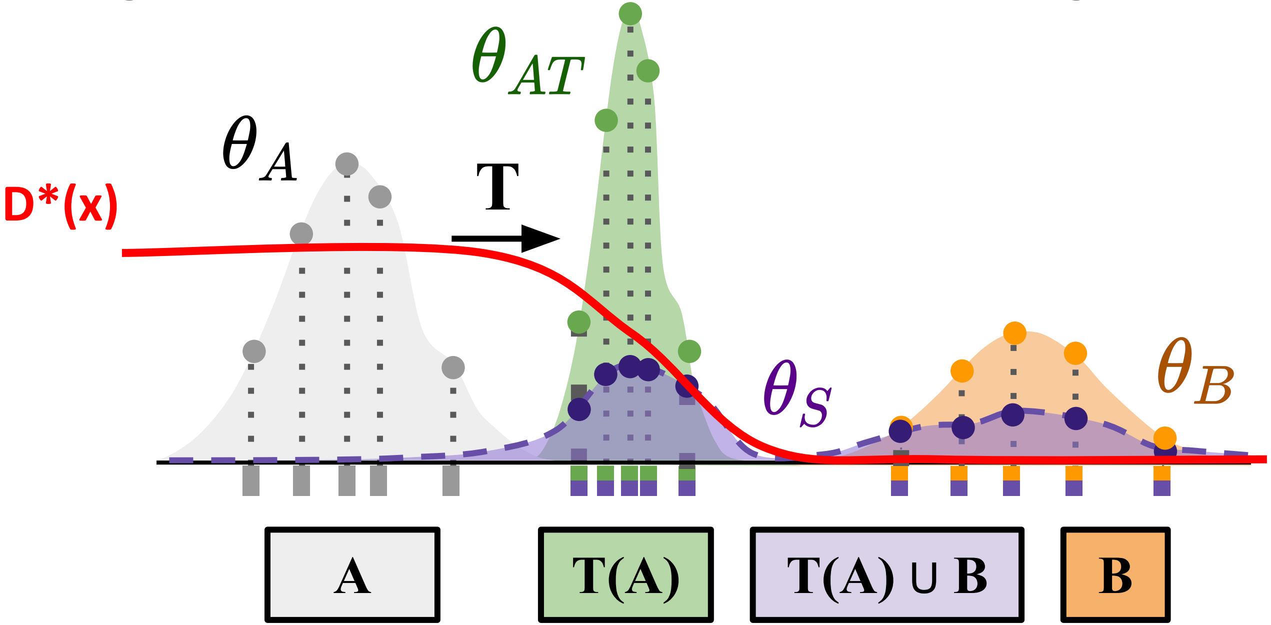
$$d_\Lambda(A, B) = 2 \cdot \mathrm{JSD}(A, B) - \mathcal{D}_{KL}(A, M) - \mathcal{D}_{KL}(B, M) + 2 \cdot \mathcal{D}_{KL}((A+B)/2, M)$$

3. Minimizing LRMF $\Leftrightarrow$ training a GAN with a particular discriminator class

$$\max_{D \in \mathcal{H}} \left[\log D(T(A)) + \log\left(1 - D(B)\right) + \log 4 \right], \quad \mathcal{H}(\theta, \theta') = \left\{ \frac{P_M(x; \theta)}{P_M(x; \theta) + P_M(x; \theta')} \right\}$$

“Log-Likelihood Ratio Minimizing Flows”, Usman, Sud, Dufour, Saenko (NeurIPS’20)

Bounding Likelihood Ratios with Normalizing Flows: Method

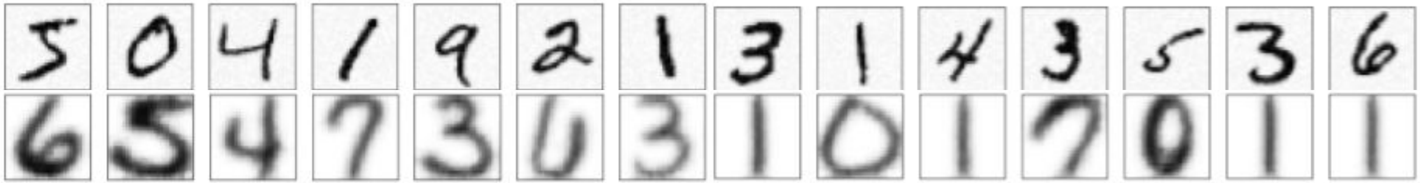


The **Optimal Bayes Classifier D*(x)** is defined in closed form

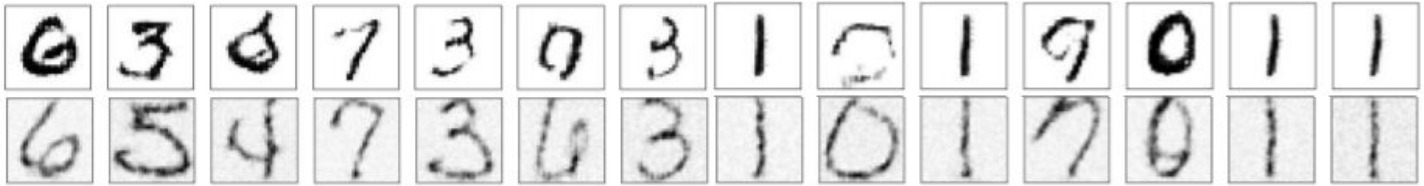
$$D^*(x) = \frac{P(X|\theta_{AT})}{P(X|\theta_{AT})+P(X|\theta_B)}$$

“Log-Likelihood Ratio Minimizing Flows”, Usman, Sud, Dufour, Saenko (NeurIPS’20)

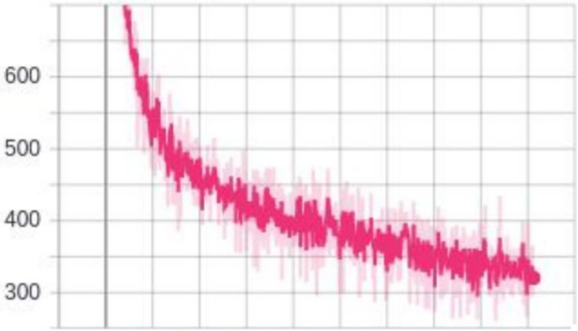
DATA
MNIST (B):
USPS (A):



LRMF T(A)
in emb space of VAEGAN:
in pixel space (GLOW):

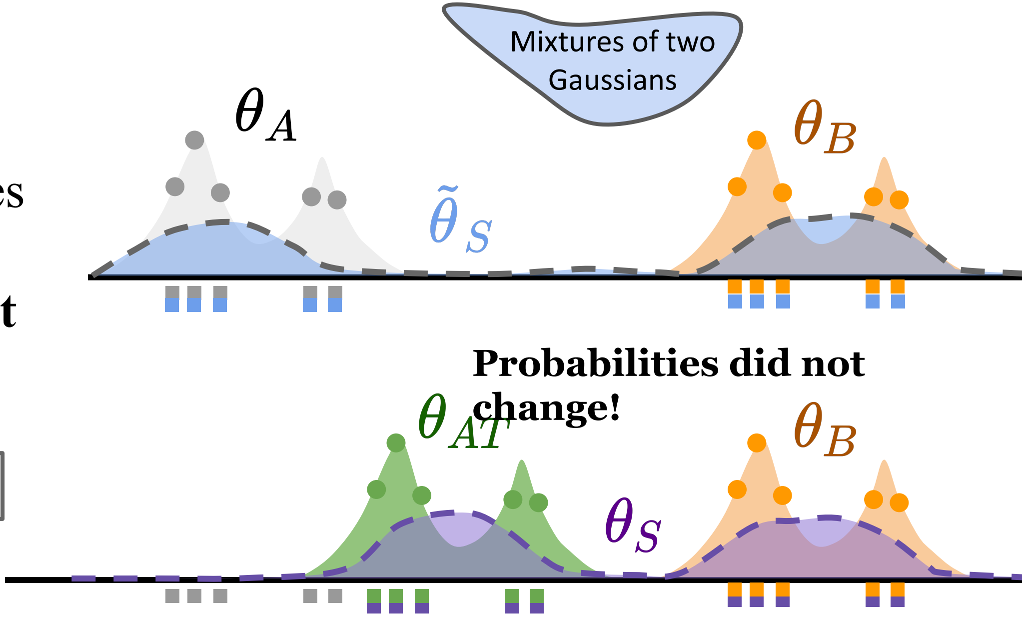


Explicit failure signal:
final LRMF loss ≠ 0



[12] baseline inspired by “AlignFlow: Cycle Consistent Learning from Multiple Domains via Normalizing Flows” by Grover et al.

Bounding Likelihood Ratios with Normalizing Flows: **Limitations**



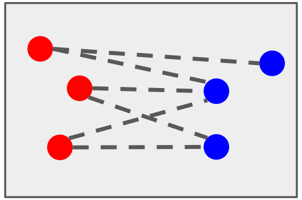
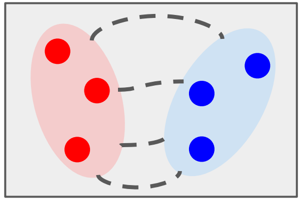
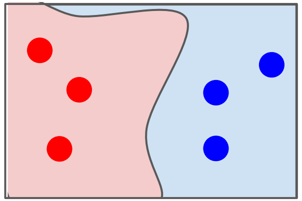
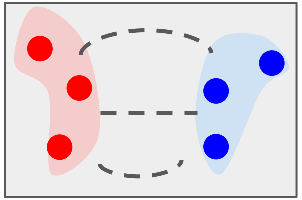
If A and B are far apart, a **shift** *T(x; b)* does not affect the likelihood of *T(A)* or *S*, so the LRMF objective is **locally constant** w.r.t. the transformation parameter *b*.

$$\boxed{||[\partial \mathcal{L}_{\text{LRMF}}(A+\mu, B, \phi, \theta)/\partial\phi]|| \propto \exp(-\mu^2)}$$

Gradients of LRMF (wrt the **transformation**) between two-gaussian mixtures **vanish** as distribution means become further away from each other.

"Log-Likelihood Ratio Minimizing Flows", Usman, Sud, Dufour, Saenko (NeurIPS'20)

Bounding Likelihood Ratios with Normalizing Flows: **Takeaway**

Non-parametric (MMD, EMD)	Simple Parametric (CORAL)	Adversarial (GAN, Monge–Kantorovich)	Log-Likelihood Ratio Minimizing Flows
model-free (metric-based)	simple data model (e.g. normal)	powerful implicit data model (NN) ✓	any tractable density + normalizing flow ✓
stable minimization ✓	stable minimization ✓	unstable min-max	stable minimization ✓
no mode collapse ✓	no mode collapse ✓	mode collapse	no mode collapse ✓
stable gradients ✓	stable gradients ✓	vanishing generator gradients	vanishing generator gradients

Learning better one-to-one mappings

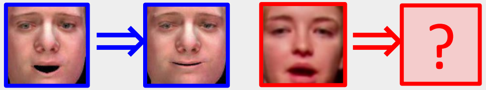
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W'18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

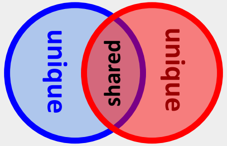
Manipulating factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)



We can manipulate attributes **unique** to each domain independently from those **shared** across domains!

(in submission)



Disclaimer

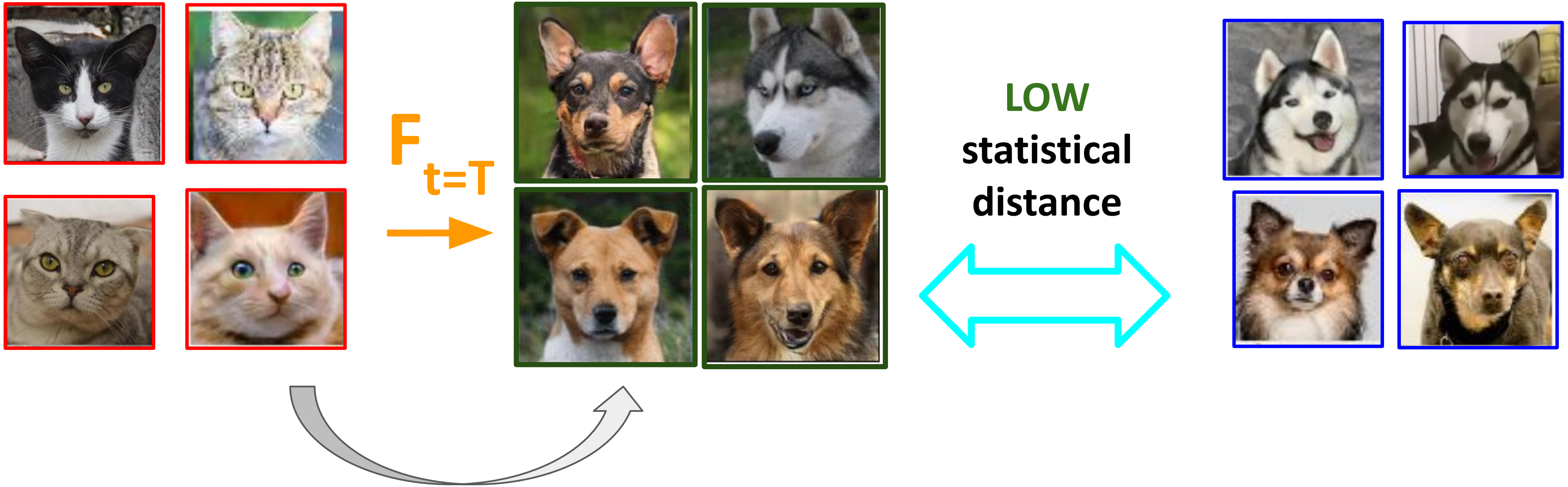
I am the <u>second</u> author, and my contribution is limited mostly to technical help:

“Adversarial Self-Defense for Cycle-Consistent GANs”,
Bashkirova, <u>Usman</u>, Saenko (NeurIPS’19)

I include this paper in this presentation, because the method proposed in this paper is essential to two remaining papers I talk about in this presentation.

What could go wrong?

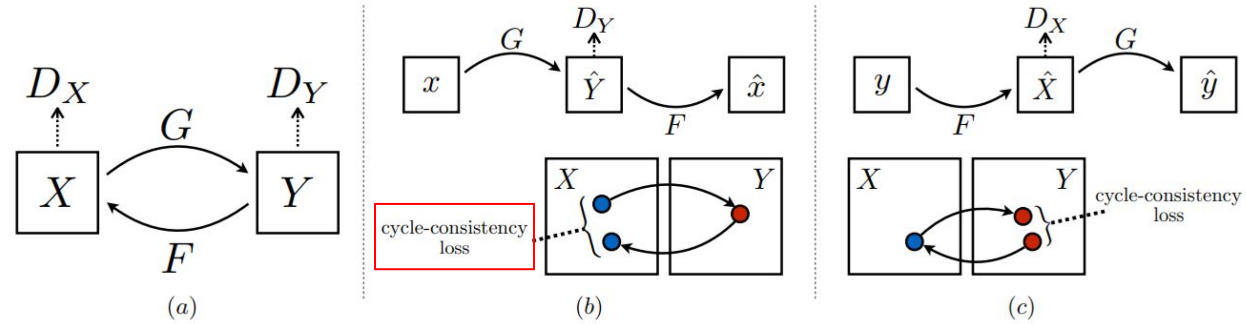
The found mapping might be <u>nonsensical</u>

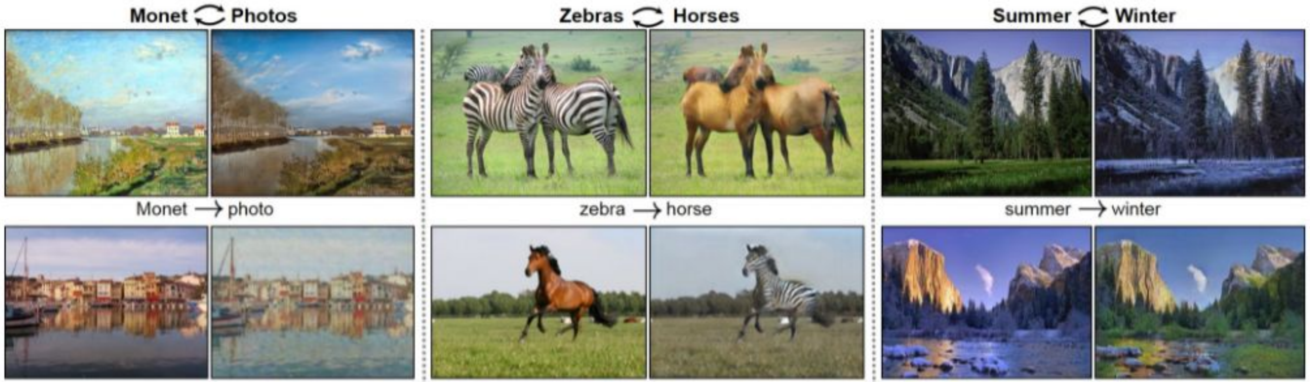


NO SEMANTIC CORRESPONDENCE

$$\min_{F \in \mathcal{F}} d(F(A), B) + R(F)$$

Cycle Reconstruction Improves Semantic Consistency

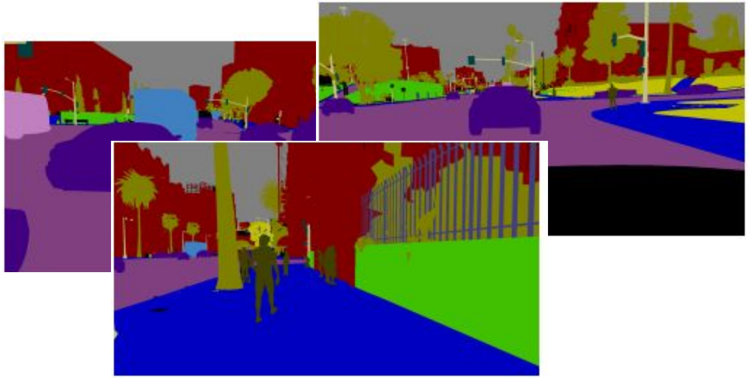




“Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”, Zhu et. al., ICCV2017

Improving Semantic Consistency with Translation Honesty

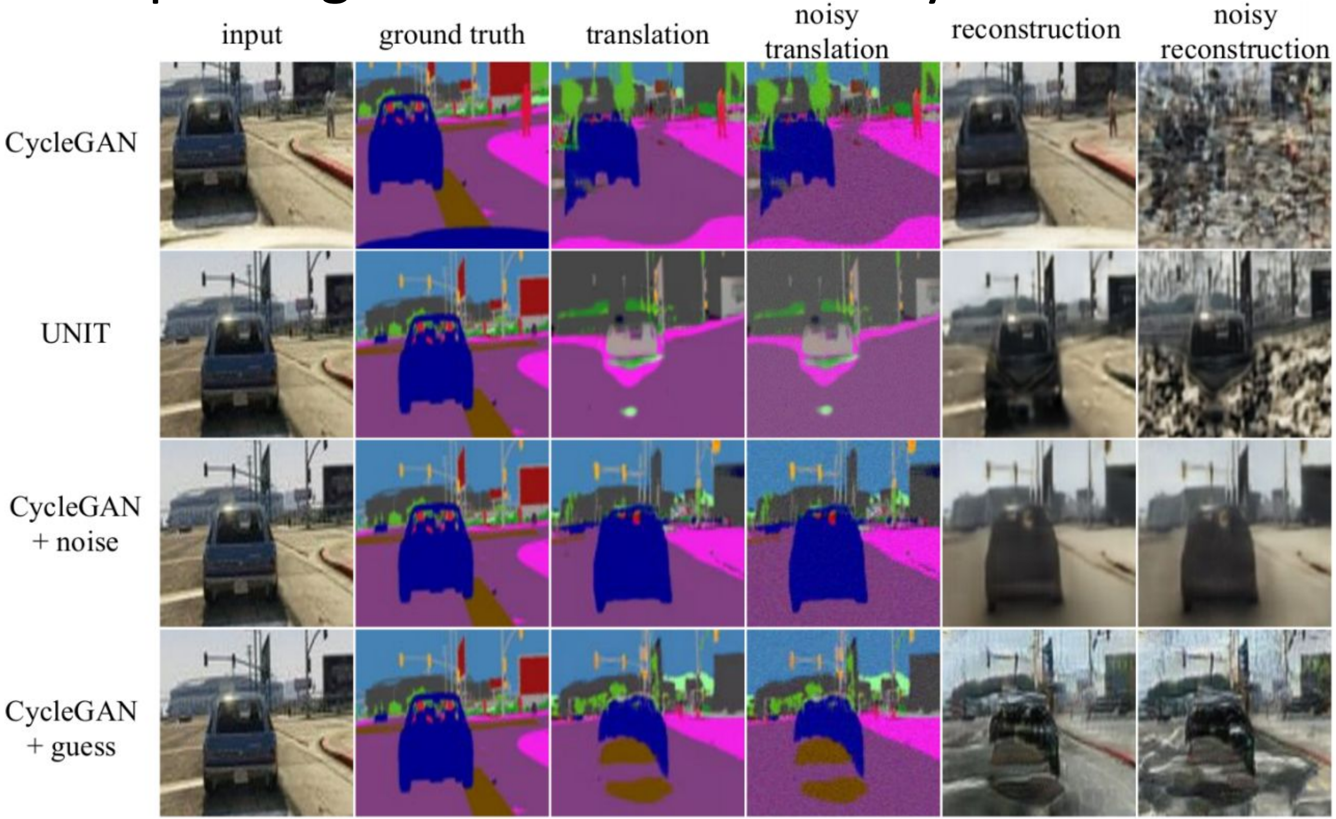
RGB **Segmentation**



CycleGAN
UNIT
…

“Adversarial Self-Defense for Cycle-Consistent GANs”, Bashkirova, Usman, Saenko (NeurIPS’19)

Improving Semantic Consistency with Translation Honesty

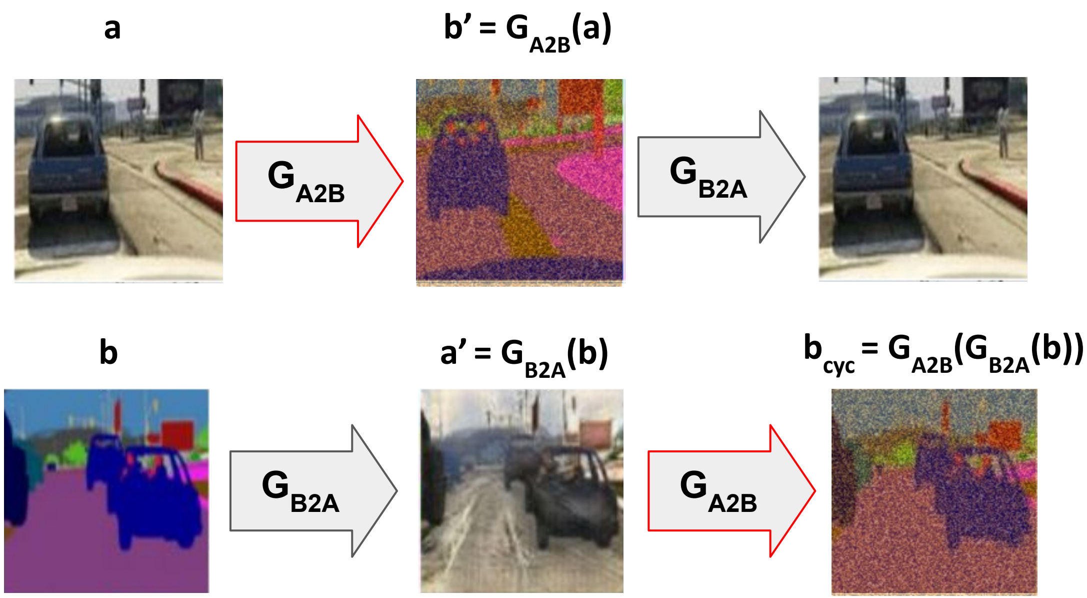


Observation: CycleGAN reconstructs input images perfectly by **cheating** - it embedded **structured noise** into generated translations.

Solution: we propose defence techniques that prevent this "cheating", and, consequently, **improve the semantic consistency** of outputs.

"Adversarial Self-Defense for Cycle-Consistent GANs", Bashkirova, Usman, Saenko (NeurIPS'19)
"CycleGAN, a Master of Steganography", Chu et al, NeurIPS'17 Workshop

Improving Semantic Consistency with Translation Honesty

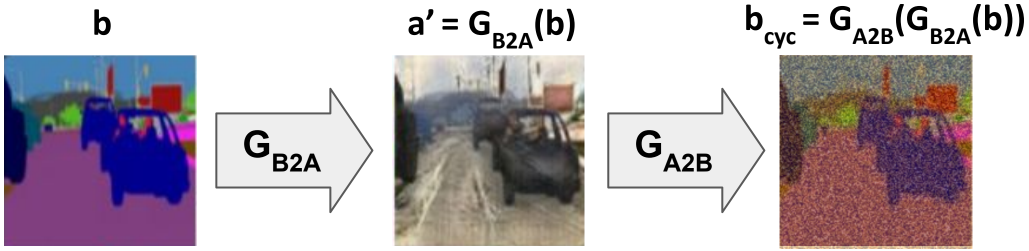


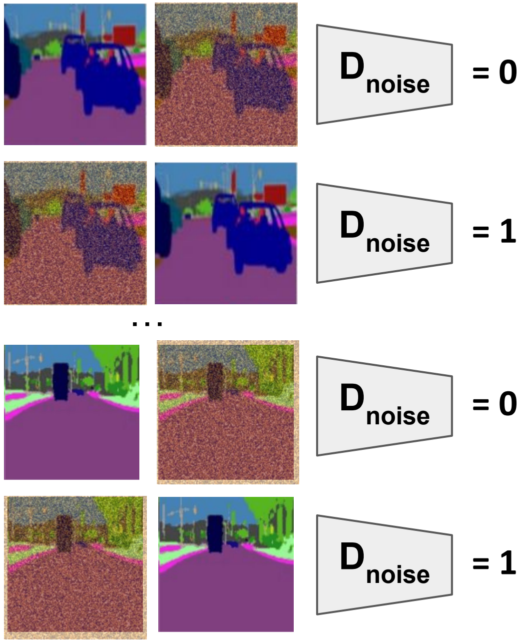
since we **know** that the network $\mathbf{G_{A2B}}$ adds adversarial noise

the cycle-reconstruction $\mathbf{b_{cyc}}$ will **also** have embedded adversarial noise!

"Adversarial Self-Defense for Cycle-Consistent GANs", Bashkirova, Usman, Saenko (NeurIPS'19)

Improving Semantic Consistency with Translation Honesty





= 0 - “the **first** image is the **original**, the **second** is a **cycle reconstruction**”

= 1 - “the **first** image is a **cycle reconstruction**, the **second** is the **original**”

Use the adversarial noise detector to penalize the model!

“Adversarial Self-Defense for Cycle-Consistent GANs”, Bashkirova, <u>Usman</u>, Saenko (NeurIPS’19)

Improving Semantic Consistency with Translation Honesty

Reconstruction honesty - how much the performance decreases if we **quantize segmentations**?

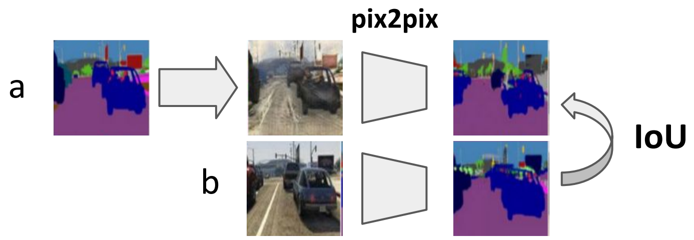
$$RH = \frac{1}{N}\sum_{i=1}^{N}\{\|G_A(\lfloor G_B(X_i)\rfloor) - Y_i\|_2 - \|G_A(G_B(X_i)) - Y_i\|_2\},$$

Sensitivity to noise - how much the output changes if we **add noise**?

$$SN(\sigma) = \frac{1}{N}\sum_{i=1}^{N}\|G_A(G_B(X_i) + \mathcal{N}(0,\sigma)) - G_A(G_B(X_i))\|_2$$

Pix2Pix IoU - do generated images produce same segmentation maps as the original?

$$\mathrm{IoU}(pix(G_A(B_i)), pix(A_i))$$



“Adversarial Self-Defense for Cycle-Consistent GANs”, Bashkirova, Usman, Saenko (NeurIPS’19)

Improving Semantic Consistency with Translation Honesty

Method	acc. segm ↑	IoU segm↑	IoU p2p↑	RH↓	SN↓
CycleGAN	0.23	0.16	0.20	27.43 ± 6.1	446.9
CycleGAN + noise*	**0.24**	0.17	0.23	9.17 ± 7.4	**94.2**
CycleGAN + guess*	0.24	**0.17**	0.21	11.4 ± 7.0	212.6
CycleGAN + guess + noise*	0.236	**0.17**	**0.24**	$\mathbf{6.1 \pm 5.9}$	150.6
UNIT	0.08	0.04	0.06	6.4 ± 11.7	361.5
MUNIT + cycle	0.13	0.08	0.17	2.5 ± 8.9	244.9
pix2pix (supervised)	0.4	0.34	–	–	–

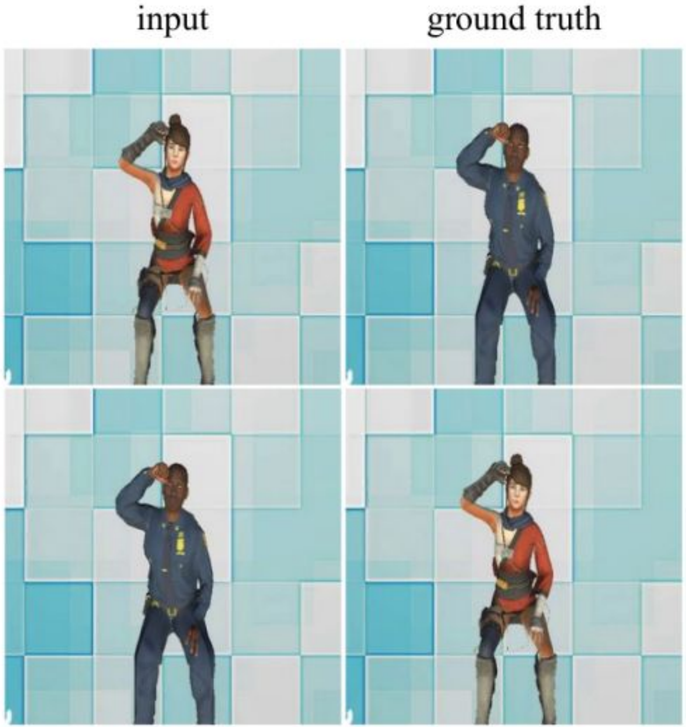
Table 2: Results on the GTA V dataset.

Method	acc. segm↑	IoU segm↑	IoU p2p↑	RH ↓	SN↓
CycleGAN	0.23	0.18	0.21	21.8 ± 5.2	251.2
CycleGAN + noise*	0.24	0.19	0.22	12.27 ± 4.42	**222.2**
CycleGAN + guess*	0.24	0.184	**0.224**	7.5 ± 2.4	235.4
CycleGAN + guess + noise*	**0.25**	**0.19**	0.22	$\mathbf{-0.45 \pm 2.3}$	238.3
UNIT	0.21	0.15	0.12	19.6 ± 6.1	528.2
MUNIT + cycle	0.15	0.09	0.12	21.4 ± 7.9	687.3
pix2pix (supervised)	0.3	0.23	–	–	–

Table 3: Results on the Google Maps dataset.

"Adversarial Self-Defense for Cycle-Consistent GANs", Bashkirova, Usman, Saenko (NeurIPS'19)

Improving Semantic Consistency with Translation Honesty



Method	MSE↓	SN ↓
CycleGAN	32.55	6.5
CycleGAN+noise*	**22.18**	**1.1**
CycleGAN+guess*	23.57	2.4
CycleGAN+guess+noise*	23.13	1.35

Table 1: Results on SynAction dataset: mean square error of the translation and sensitivity to noise.

“Adversarial Self-Defense for Cycle-Consistent GANs”, Bashkirova, Usman, Saenko (NeurIPS’19)

Improving Semantic Consistency with Translation Honesty: **Takeaway**

Cycle-consistent models hide information in the form of adversarial noise.

If we prevent them from doing this, the semantic consistency of the alignment improves.

Learning better one-to-one mappings

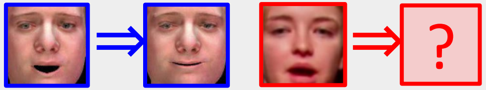
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W'18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

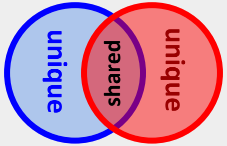
Manipulating factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)

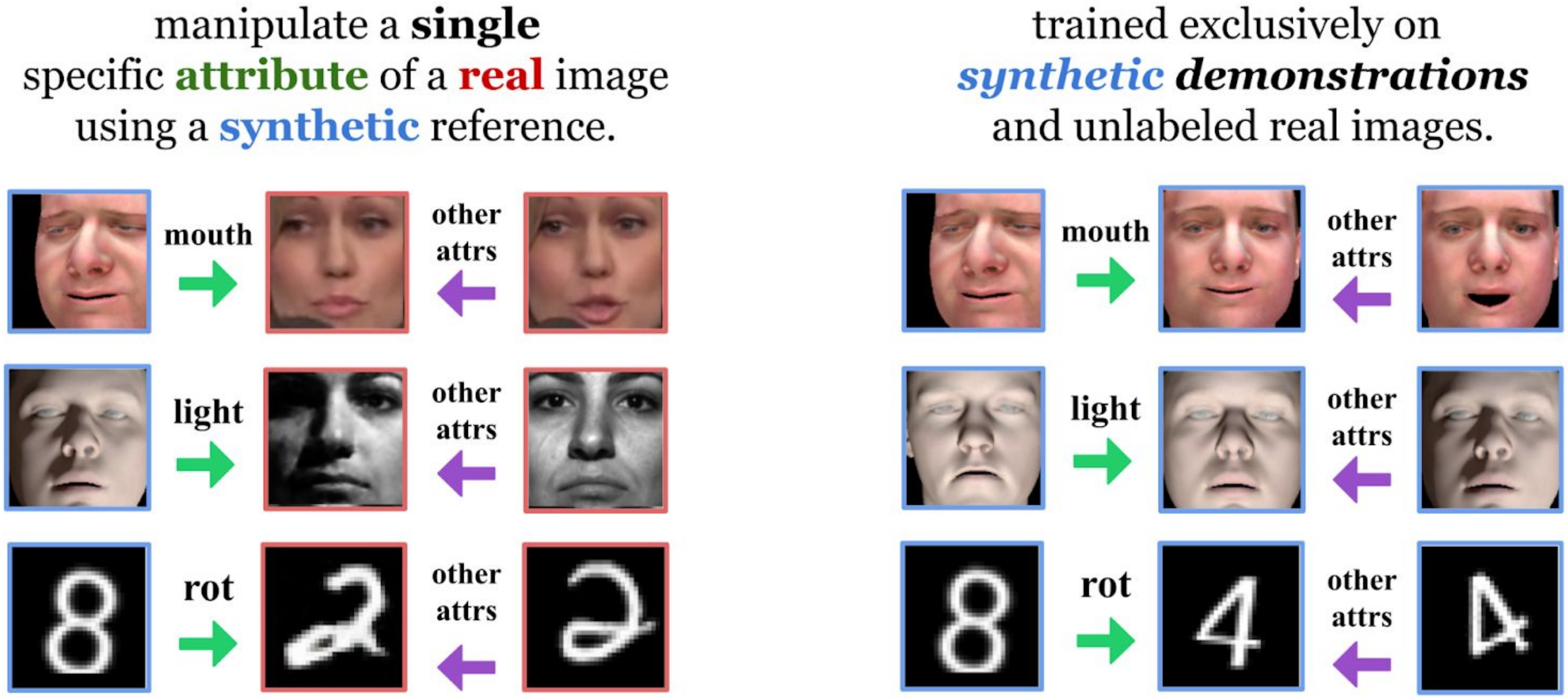


We can manipulate attributes **unique** to each domain independently from those **shared** across domains!

(in submission)



Learning from Cross-Domain Demonstrations: Task

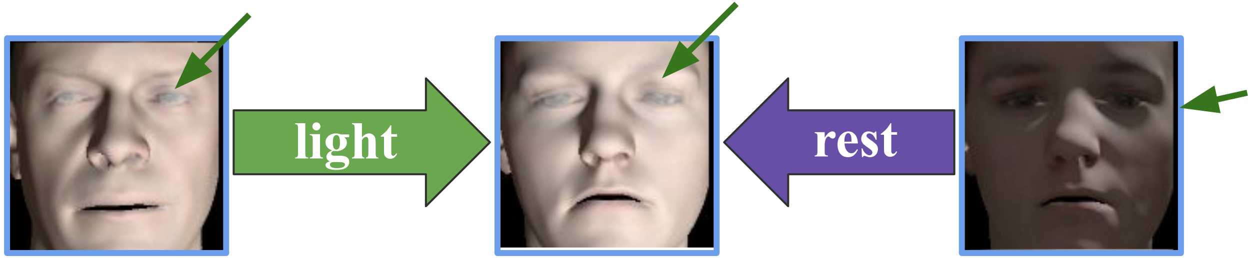


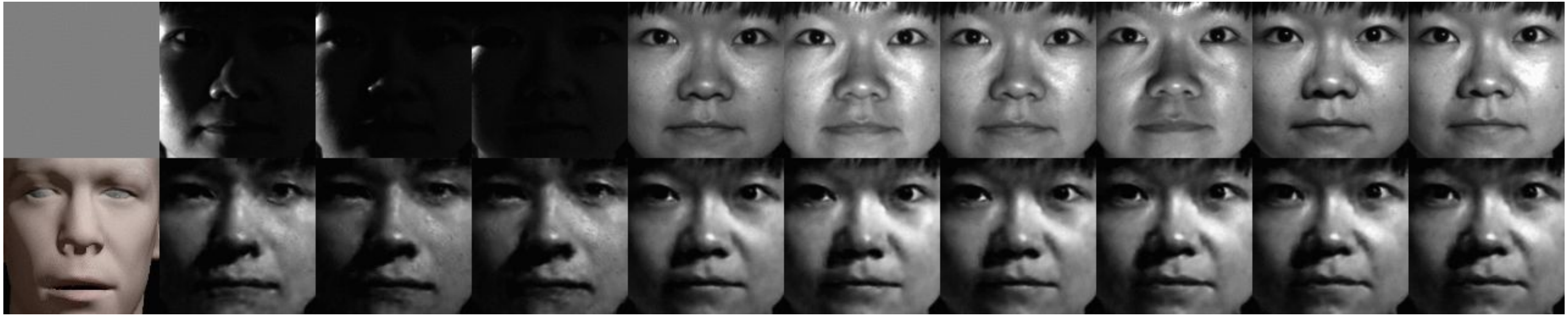
"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

Learning from Cross-Domain Demonstrations: Demo

"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

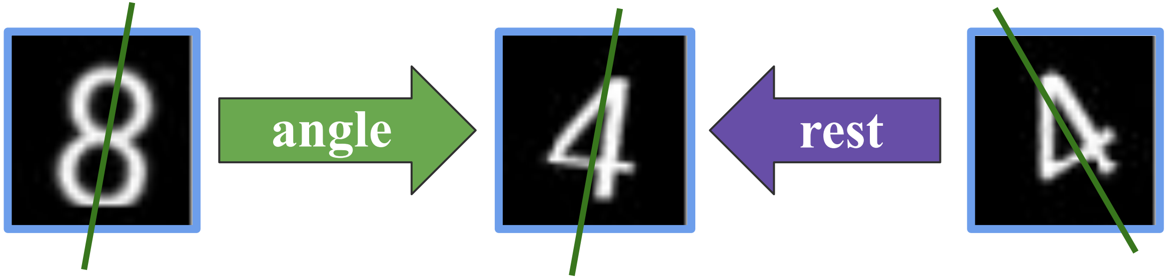
Learning from Cross-Domain Demonstrations: Demo

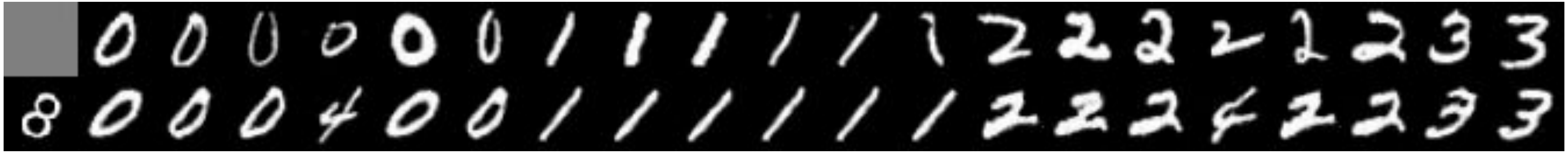




“PuppetGAN: Cross-Domain Image Manipulation by Demonstration” by Usman, Dufour, Saenko, Bregler (ICCV’19)

Learning from Cross-Domain Demonstrations: Demo





“PuppetGAN: Cross-Domain Image Manipulation by Demonstration” by Usman, Dufour, Saenko, Bregler (ICCV’19)

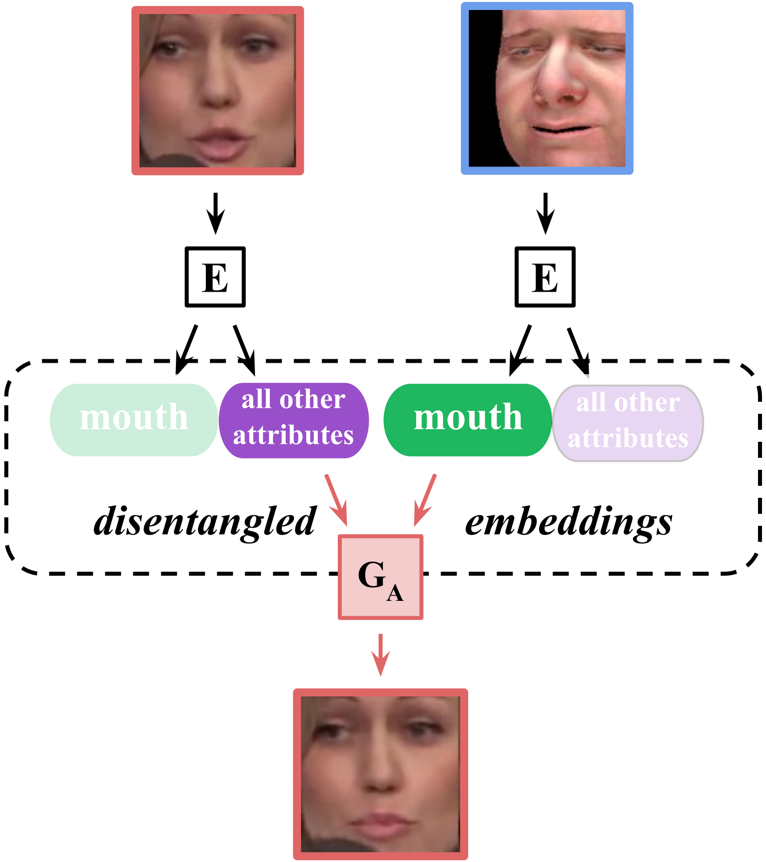
PuppetGAN: Method

Our goal is to train a model that **splits** the embedding into **two parts:**

- **one** to represent the attribute we manipulate (mouth),
- the **other** to represent all other attributes (hair, mic, etc).

real
decoder

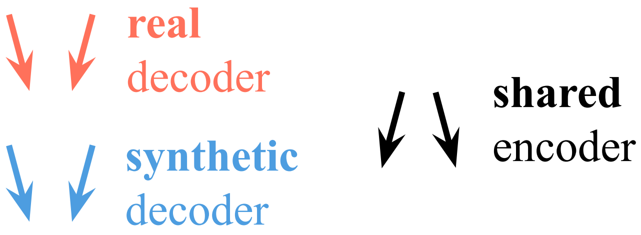
shared
encoder

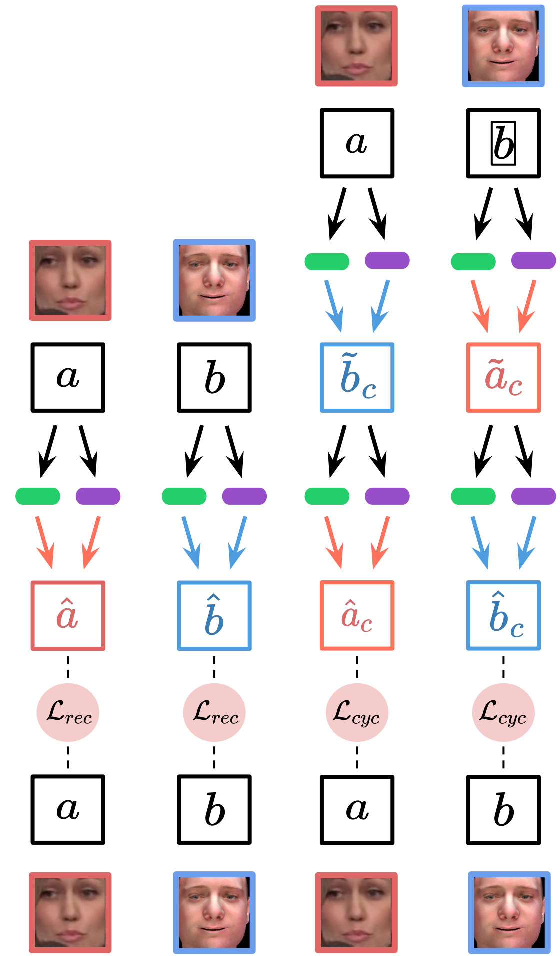


“PuppetGAN: Cross-Domain Image Manipulation by Demonstration” by Usman, Dufour, Saenko, Bregler (ICCV’19)

PuppetGAN: Method

We used **autoencoder**
and **cycle losses** on both domains.

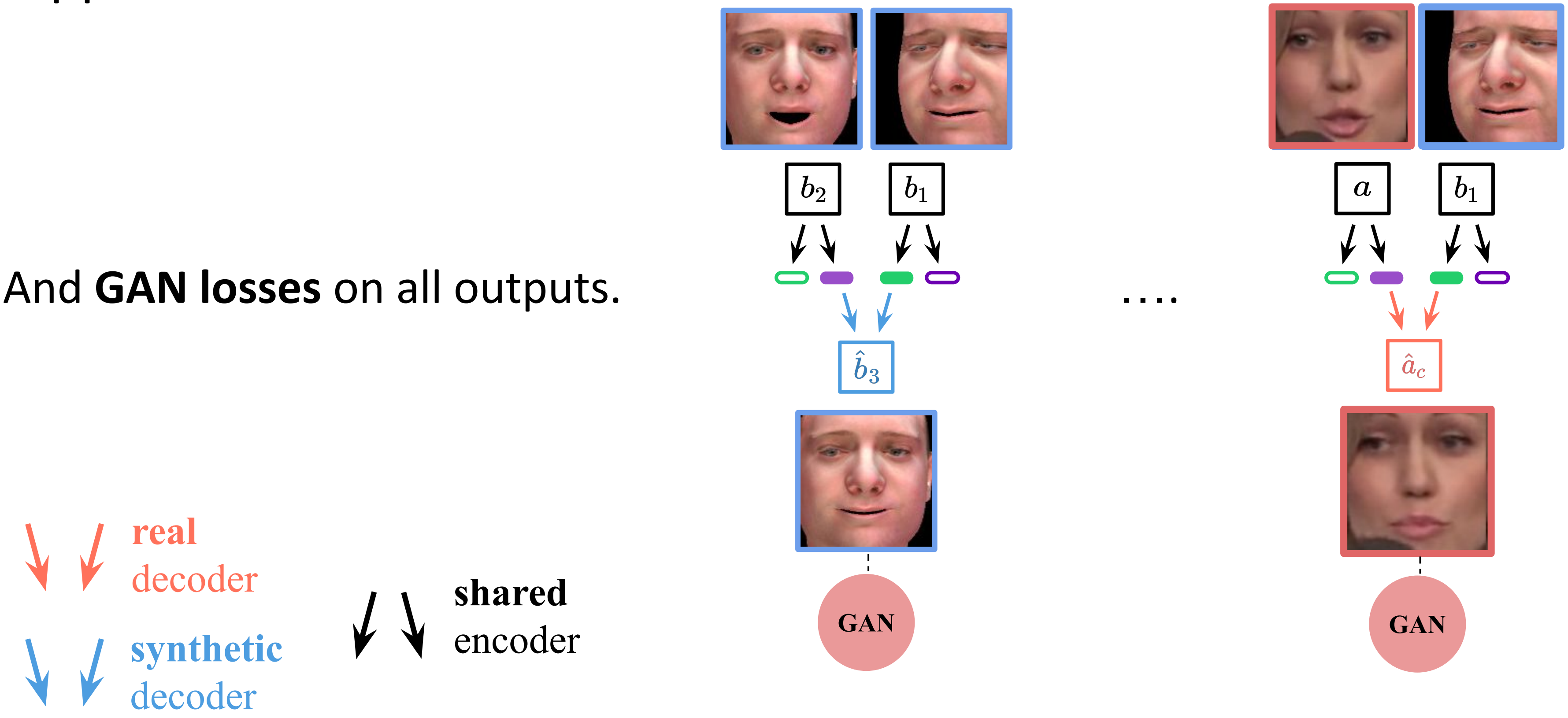




"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by <u>Usman</u>, Dufour, Saenko, Bregler (ICCV'19)

PuppetGAN: Method

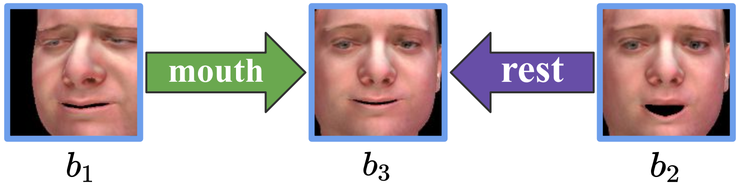
And **GAN losses** on all outputs.

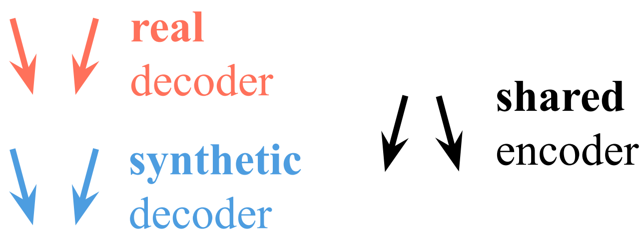


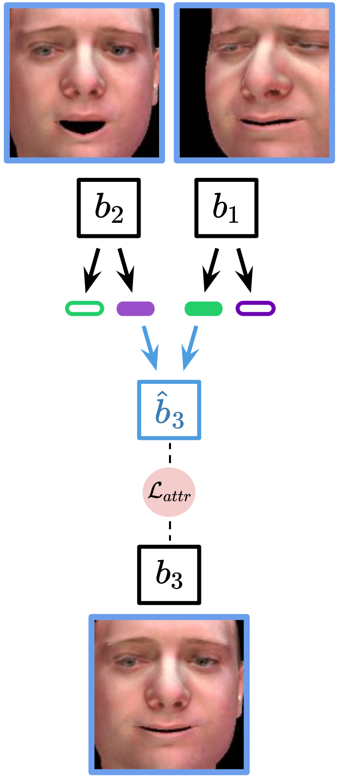
"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

PuppetGAN: Method

We used **supervised losses** on synthetic data.







"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

PuppetGAN: Method

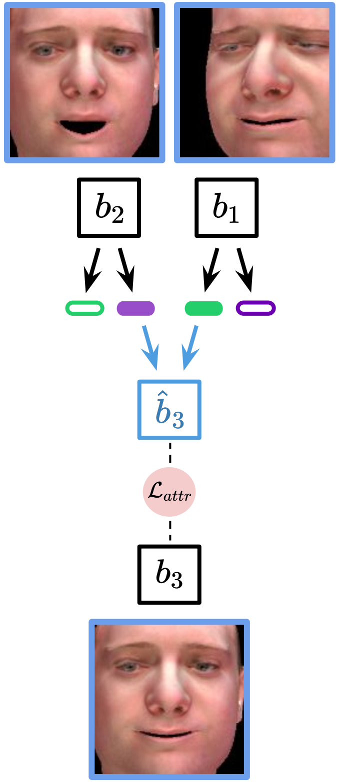
Problem:

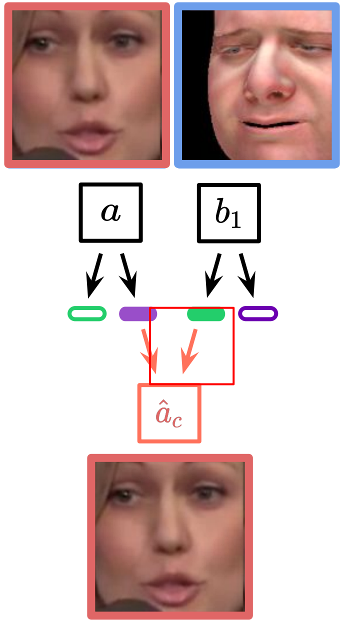
The real decoder might **ignore** one input.

real decoder

synthetic decoder

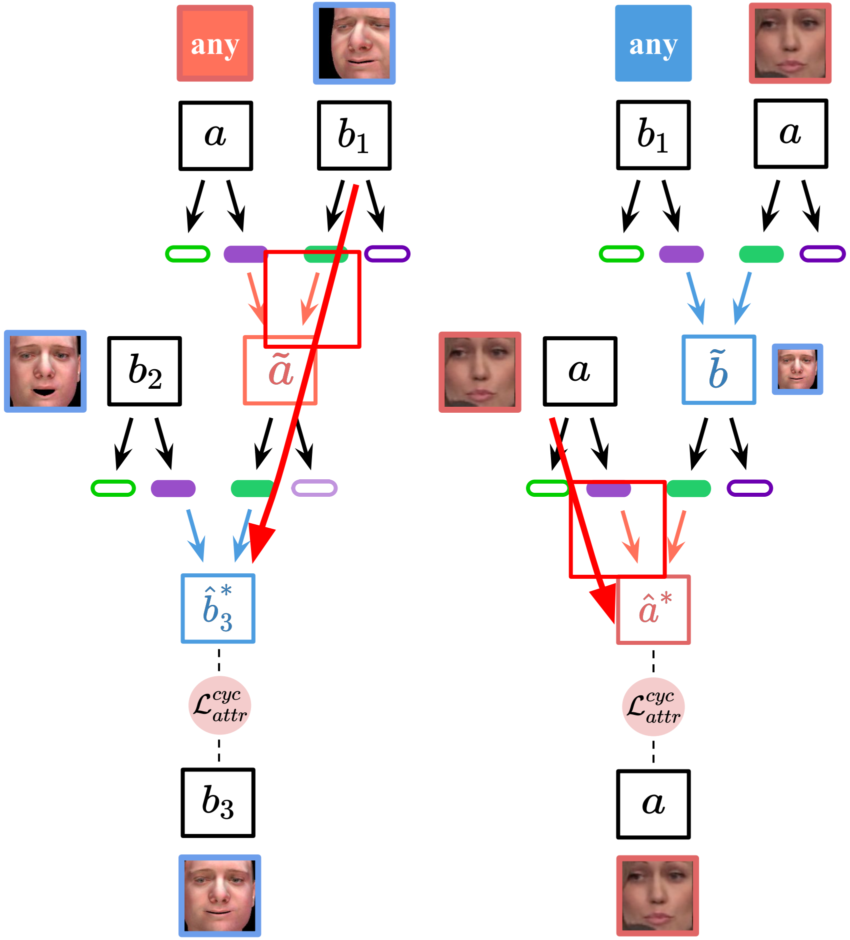
shared encoder

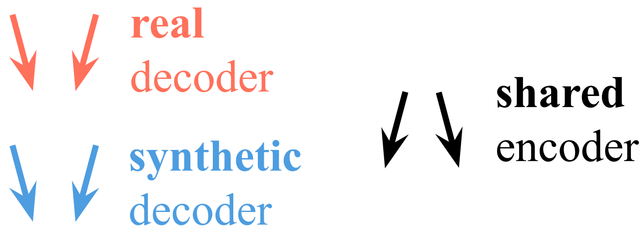




PuppetGAN: Method

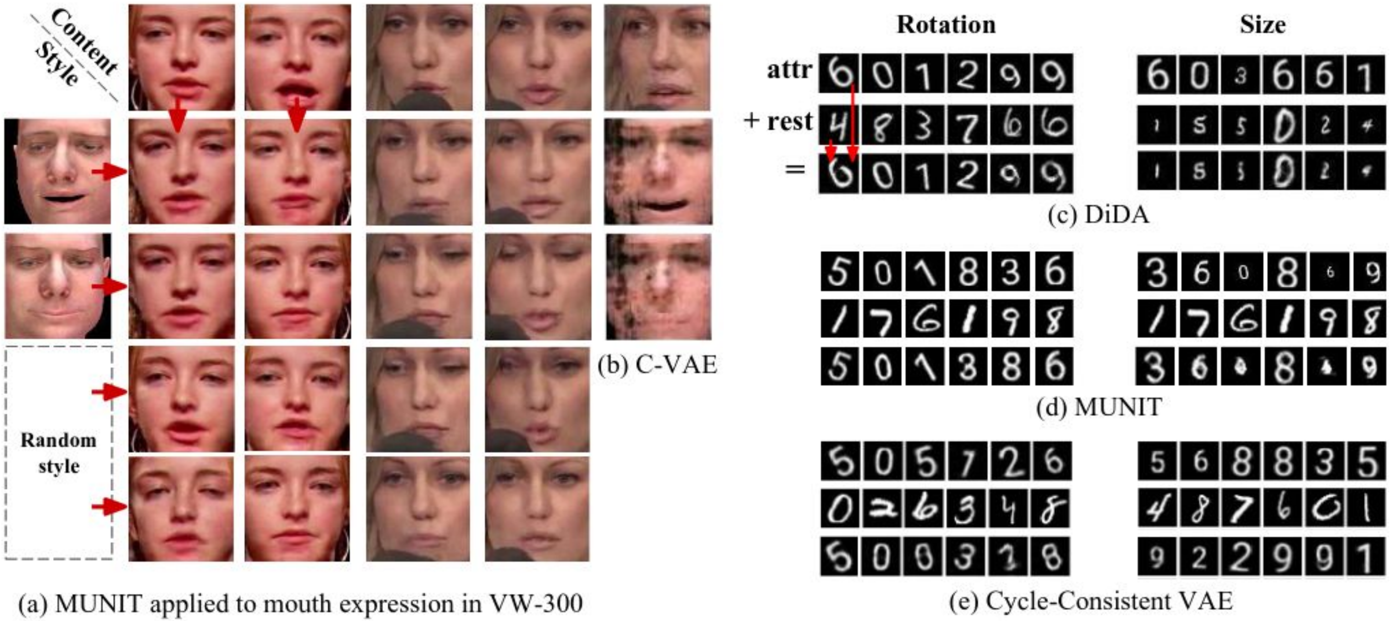
We used **compositional constraint losses** to ensure that **all embeddings** are used.





"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

PuppetGAN: Comparing to related work



“PuppetGAN: Cross-Domain Image Manipulation by Demonstration” by <u>Usman</u>, Dufour, Saenko, Bregler (ICCV’19)

PuppetGAN: Comparing to related work

Model	Disentanglement Quality								Input Domain Discrepancy			
	Size				Rotation				Size		Rot	
	Acc $\uparrow$	$r_{\text{attr}}^{\text{syn}} \uparrow$	$r_{\text{rest}}^{\text{syn}} \downarrow$	$V_{\text{rest}} \downarrow$	Acc $\uparrow$	$r_{\text{attr}}^{\text{syn}} \uparrow$	$r_{\text{rest}}^{\text{syn}} \downarrow$	$V_{\text{rest}} \downarrow$	$J_{\text{attr}}^{\text{syn}}$	$J_{\text{rest}}^{\text{syn}}$	$J_{\text{attr}}^{\text{syn}}$	$J_{\text{rest}}^{\text{syn}}$
PuppetGAN	**0.73**	**0.85**	**0.02**	**0.02**	**0.97**	**0.40**	**0.11**	**0.01**				
CycleGAN [28]	0.10	0.28	**0.06**	0.28	0.11	**0.54**	0.37	0.33				
DiDA [2]	**0.71**	0.18	0.09	**0.02**	**0.86**	0.04	0.35	**0.02**	0.27	0.78	0.05	2.20
MUNIT [10]	**0.96**	0.06	0.09	**0.01**	**1.00**	0.00	0.15	**0.01**				
Cycle-VAE [8]	0.17	**0.92**	0.16	**0.01**	0.29	**0.45**	**0.10**	**0.01**				
PuppetGAN†	**0.64**	0.28	0.07	**0.01**	0.10	0.06	**0.04**	**0.01**	0.90	0.92	0.06	108

"PuppetGAN: Cross-Domain Image Manipulation by Demonstration" by Usman, Dufour, Saenko, Bregler (ICCV'19)

PuppetGAN: **Takeaway**

We can manipulate specific attributes of real images using supervision from crude synthetic simulations!

Learning better one-to-one mappings

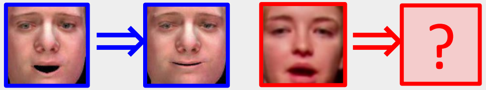
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W’18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

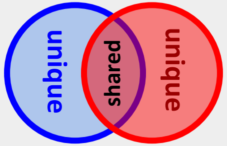
Manipulating factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)

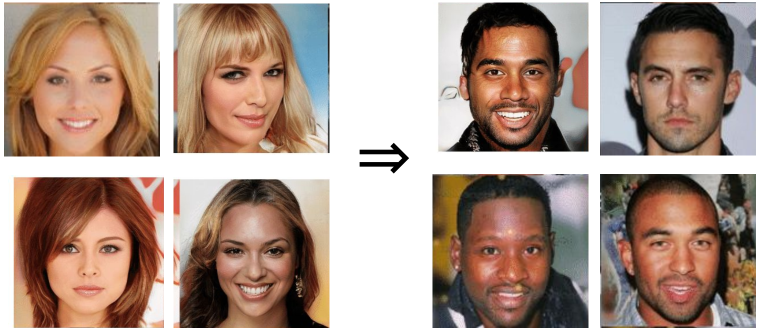


We can manipulate attributes **unique** to each domain independently from those **shared** across domains!

(in submission)



Disentangling Domain-Specific and Domain-Invariant Factors of Variation



1-to-1 alignment problem
is not well defined!

Preserve **as much**
as possible

many-to-many alignment problem is well defined!

Disentangling Domain-Specific and Domain-Invariant Factors of Variation

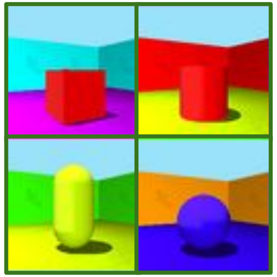
Goal 1: <u>learn</u> which factors of variation are shared vs domain-specific <u>from data</u>

source (brunet males)

⇒

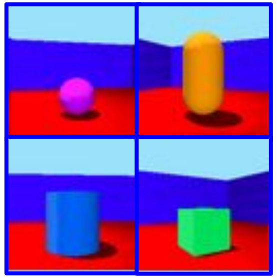
target (females)

source



⇒

target



(source-specific)
mustache

(shared)
skin tone, pose, background color

(target-specific)
hair color

(source-specific)
floor, wall colors

(shared)
object color, shape

(target-specific)
rotation and size

Goal 2: <u>translate</u> a source image to the target domain <u>guided</u> by a target example

shared from source

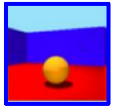
+

specific from guide ⇒

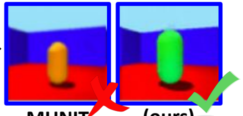
StarGANv2 (ours)

shared from source

+



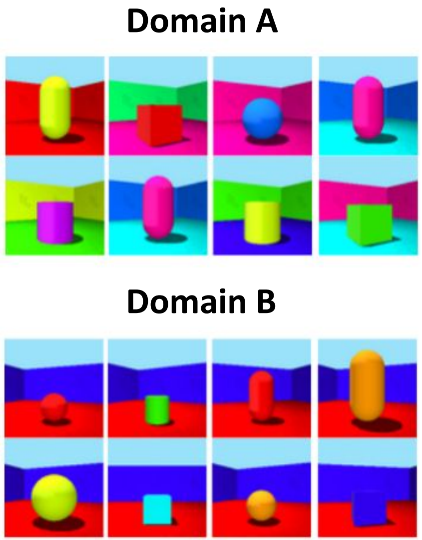
specific from guide ⇒



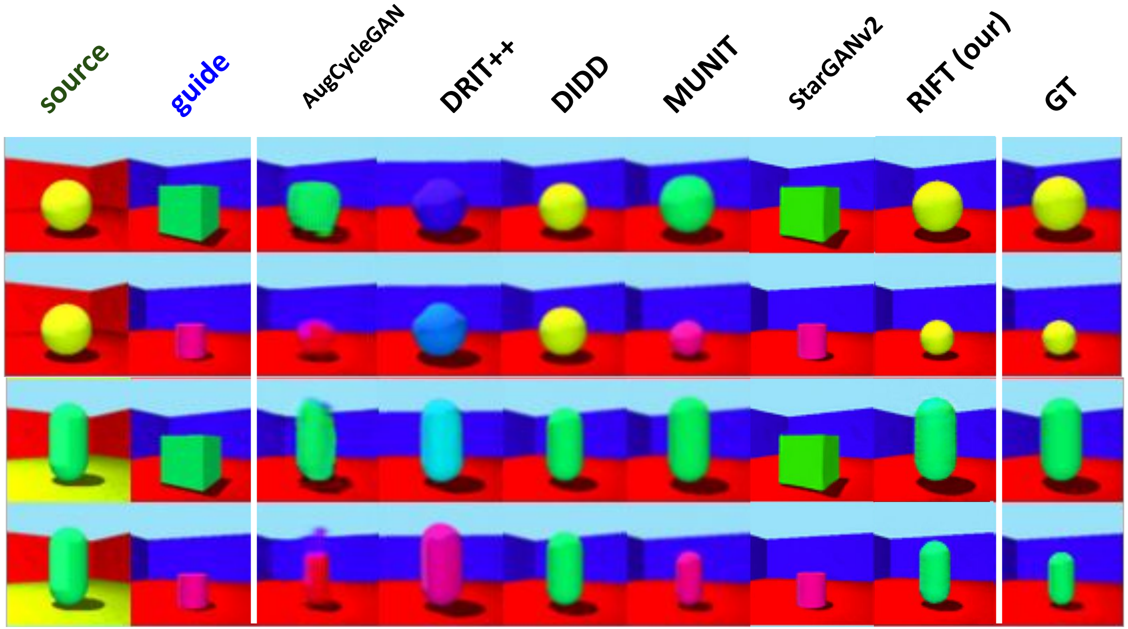
MUNIT (ours)

“RIFT: Disentangled Unsupervised Image Translation via Restricted Information Flow” by <u>Usman*</u>, Bashkirova*, Saenko (in submission)

Disentangling Domain-Specific and Domain-Invariant Factors of Variation

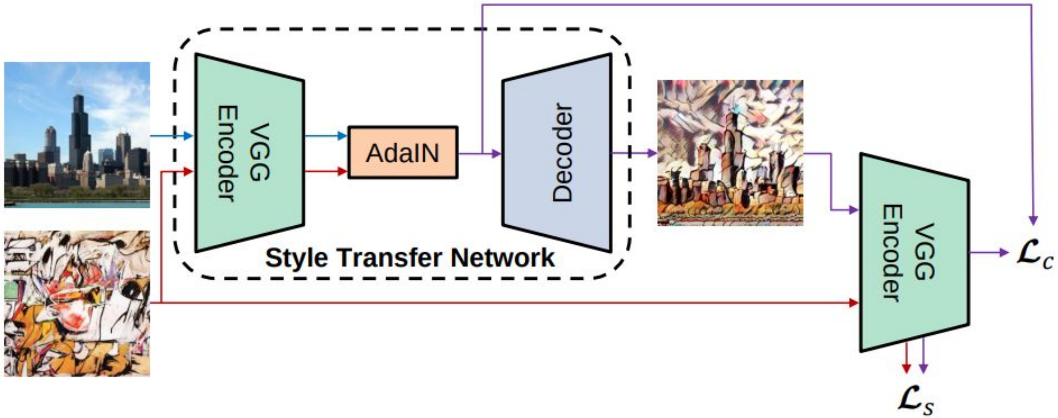


shared: object color, shape
source: floor, wall color
target: size, orientation



"Evaluation of Correctness in Unsupervised Many-to-Many Image Translation" by Bashkirova, Usman, Saenko (WACV22)
"RIFT: Disentangled Unsupervised Image Translation via Restricted Information Flow" by Usman*, Bashkirova*, Saenko (in submission)

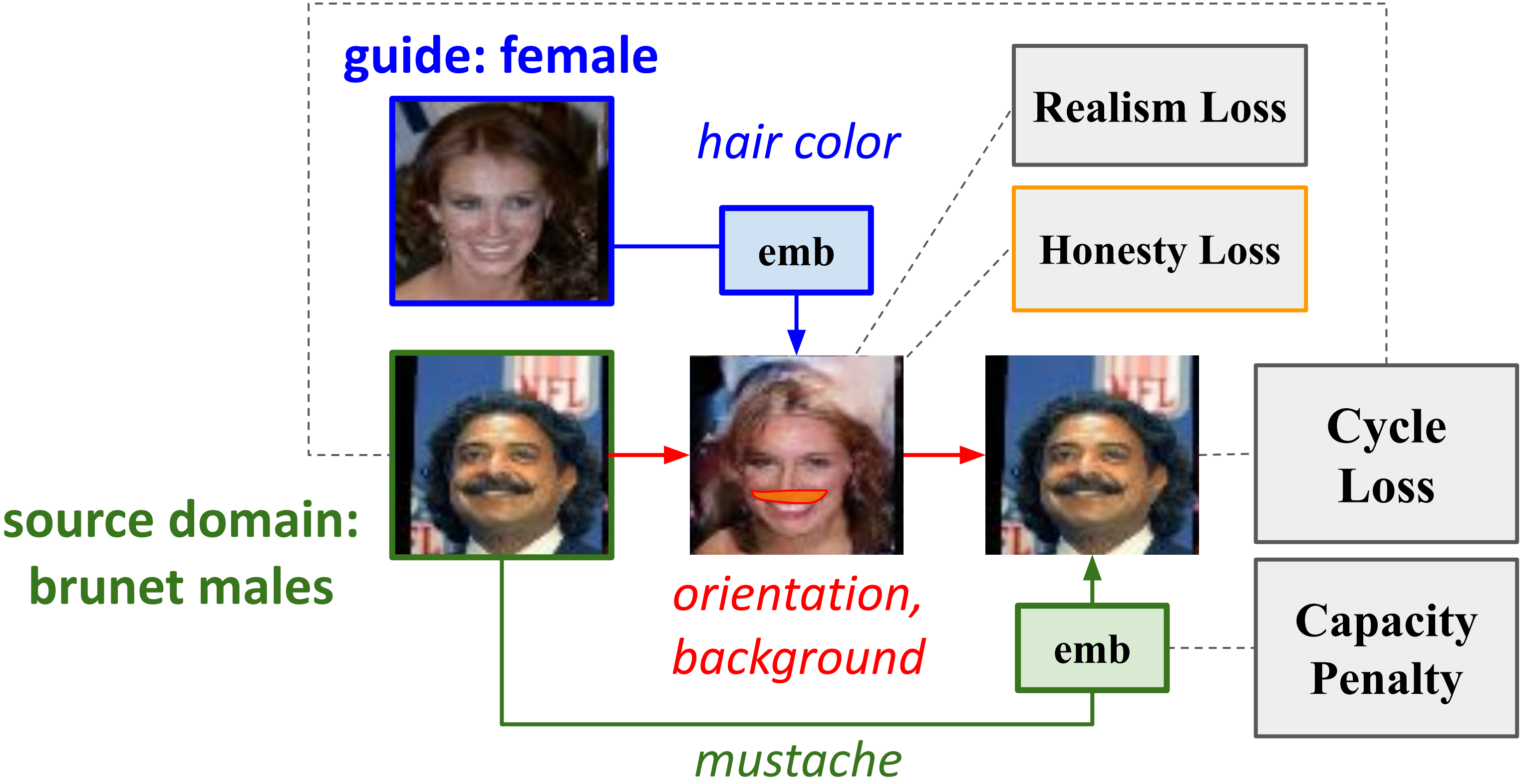
Disentangling Domain-Specific and Domain-Invariant Factors of Variation



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

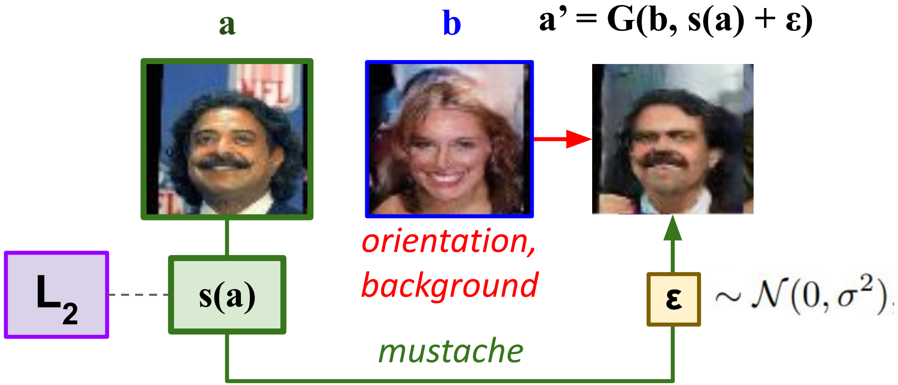
“Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization” by Huang et al., ICCV’17

RIFT: Translation via Restricted Information Flow



"RIFT: Disentangled Unsupervised Image Translation via Restricted Information Flow" by Usman*, Bashkirova*, Saenko (in submission)
"Adversarial Self-Defense for Cycle-Consistent GANs", Bashkirova, Usman, Saenko (NeurIPS'19)

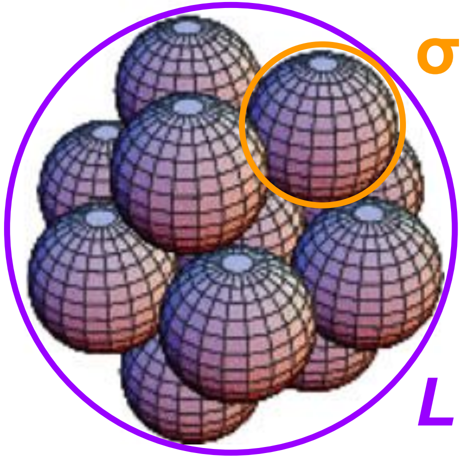
Capacity Loss



Theorem 1. *The effective capacity of the guided embedding,* i.e. *the capacity of the $a \to a'$ channel,* i.e. *the mutual information* $\mathrm{MI}(a; a')$ *is bounded by:*

$$\mathrm{MI}(a; a') \lesssim \dim(s(a)) \cdot \log_2 \left(1 + L/\sigma^2\right),$$
$$\textit{where } a' = G(b, s(a) + \varepsilon),\ \varepsilon \sim \mathcal{N}(0, \sigma^2),$$
$$\textit{and } L = \mathbb{E}\|s(a)\|_2^2,\ a \sim A,\ b \sim B$$

Corollary:
$L \to \infty \Rightarrow \mathrm{MI} \to \infty$
$\sigma \to 0 \Rightarrow \mathrm{MI} \to \infty$



"RIFT: Disentangled Unsupervised Image Translation via Restricted Information Flow" by Usman*, Bashkirova*, Saenko (in submission)

Capacity Loss

Theorem 1. *The effective capacity of the guided embedding, i.e. the capacity of the $a \to a'$ channel, i.e. the mutual information* $\mathrm{MI}(a; a')$ *is bounded by:*

$$\mathrm{MI}(a; a') \lesssim \dim(s(a)) \cdot \log_2\left(1 + L/\sigma^2\right),$$
$$\textit{where } a' = G(b, s(a) + \varepsilon),\ \varepsilon \sim \mathcal{N}(0, \sigma^2),$$
$$\textit{and } L = \mathbb{E}\|s(a)\|_2^2,\ a \sim A,\ b \sim B$$

Proof. Applying the data processing inequality

$$X \to Y \to Z \quad \Rightarrow \quad \mathrm{MI}(X; Z) \le \mathrm{MI}(X; Y) \ \wedge\ \mathrm{MI}(X; Z) \le \mathrm{MI}(Y; Z)$$

twice to following Markov chains

$$a \to (s(a) + \varepsilon) \to a', \quad a \to s(a) \to (s(a) + \varepsilon)$$

gives us

$$\mathrm{MI}(a; a') \le \mathrm{MI}(a; s(a) + \varepsilon) \le \mathrm{MI}(s(a); s(a) + \varepsilon)$$

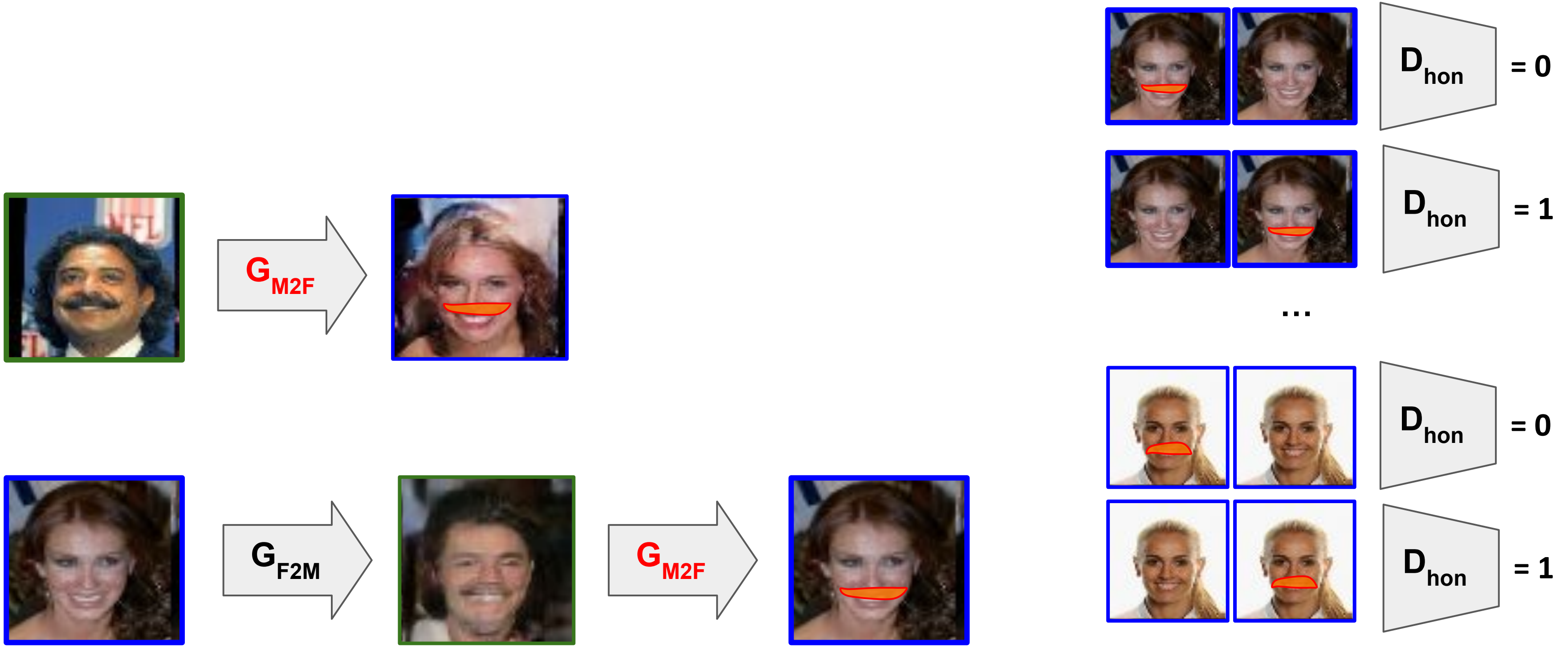
intuitively meaning that the overall pipeline always looses at least as much information as each of its steps. Then expanding the mutual information in terms of the differential entropy $h(X)$ gives us

$$\begin{aligned}\mathrm{MI}(s(a); s(a) + \varepsilon) &= h(s(a) + \varepsilon) - h(s(a) + \varepsilon | s(a)) \\ &= h(s(a) + \varepsilon) - h(\varepsilon)\end{aligned}$$

Since the the second raw moment (aka power) of $s(a)$ is bounded by L, the entropy $h(s(a) + \varepsilon)$ will be maximized if $s(a)$ is a k-dimensional spherical multivariate normal with variance L, where $k = \dim(s(a))$ therefore

$$\begin{aligned}\mathrm{MI}(s(a); s(a) + \varepsilon) &\le h(\mathcal{N}_k(0; L + \sigma^2)) + h(\mathcal{N}_k(0; \sigma^2)) \\ &= \frac{1}{2} \ln\left(\frac{(L + \sigma^2)^k}{\sigma^{2k}}\right) \le k \cdot \log_2\left(1 + L/\sigma^2\right).\end{aligned}$$

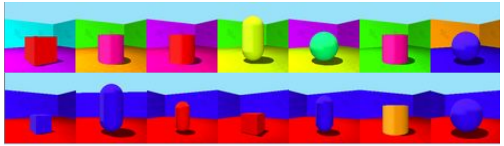
Honesty Loss



"RIFT: Disentangled Unsupervised Image Translation via Restricted Information Flow" by Usman*, Bashkirova*, Saenko (in submission)
"Adversarial Self-Defense for Cycle-Consistent GANs", Bashkirova, Usman, Saenko (NeurIPS'19)

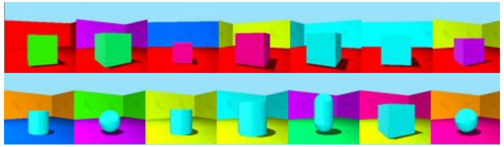
Datasets

Shapes-3D-A



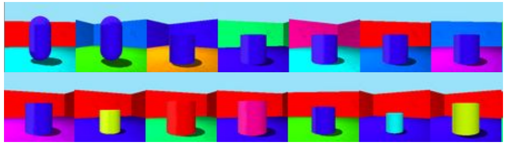
shared: object color, shape
source: floor, wall color
target: size, orientation

Shapes-3D-B



shared: wall color, size
source: object color, orient.
target: shape, floor color

Shapes-3D-C



shared: floor color, orient.
source: wall color, shape
target: size, object color

SynAction

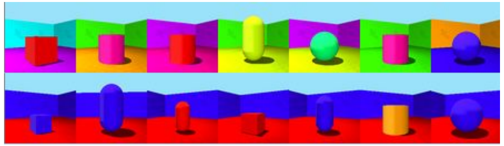
shared: pose
source: background
target: identity/clothing

CelebA

shared: pose, background
source: hair color
target: facial hair

Results

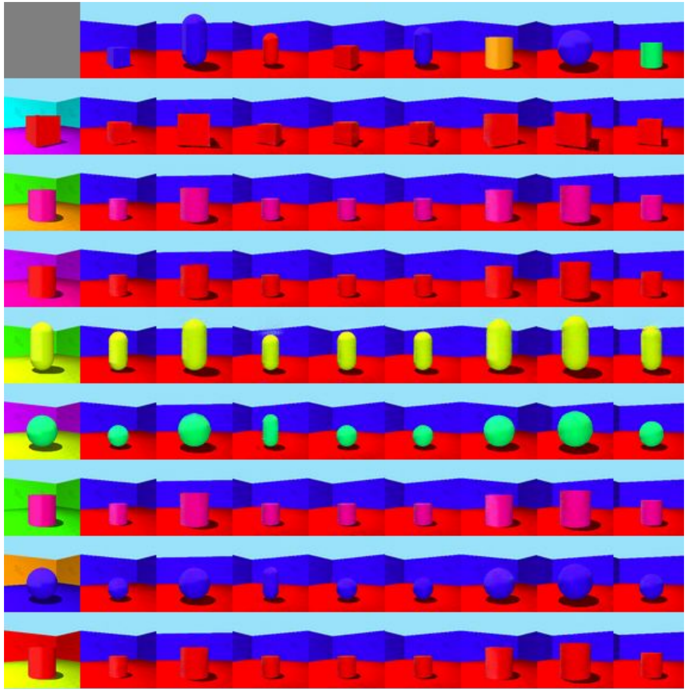
Shapes-3D-A



shared: object color, shape
source: floor, wall color
target: size, orientation

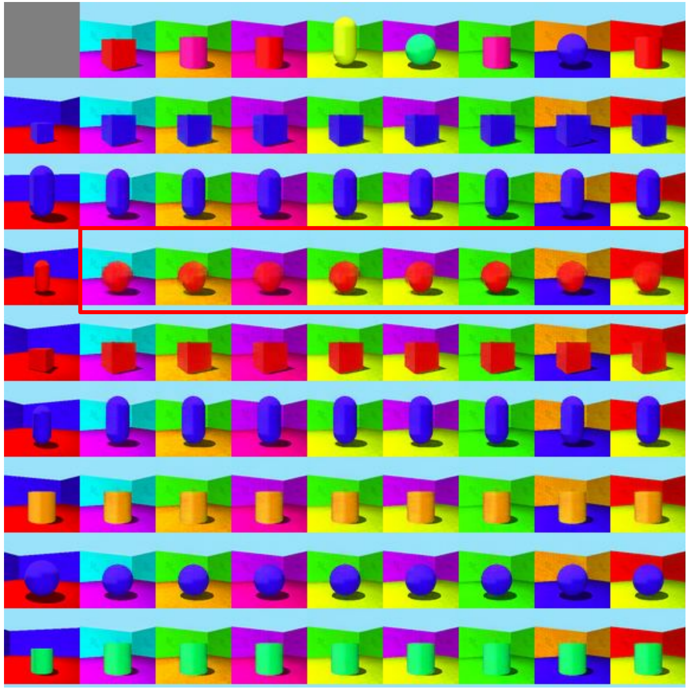
guide (rotation and size)

source (object color and shape)



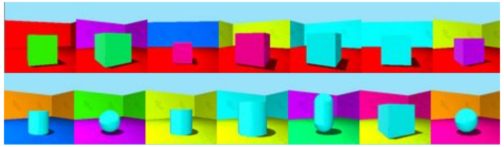
guide (floor and wall color)

source (object color and shape)

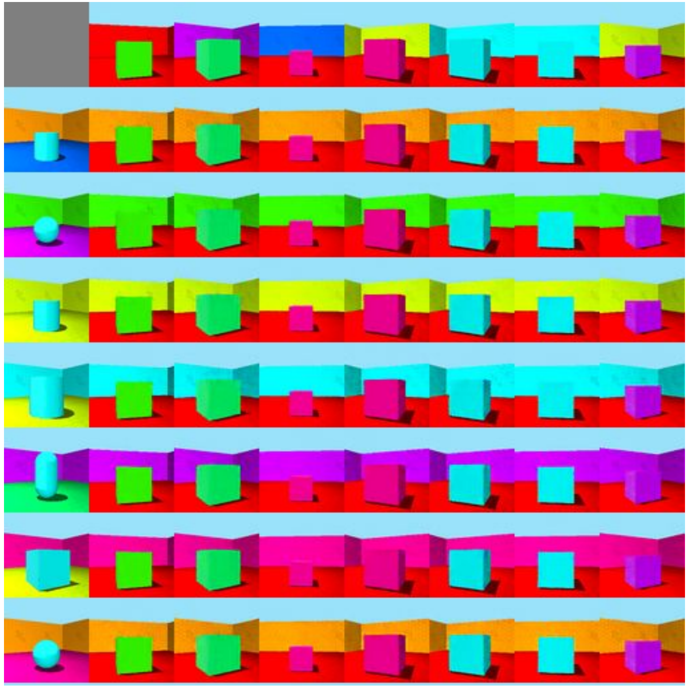


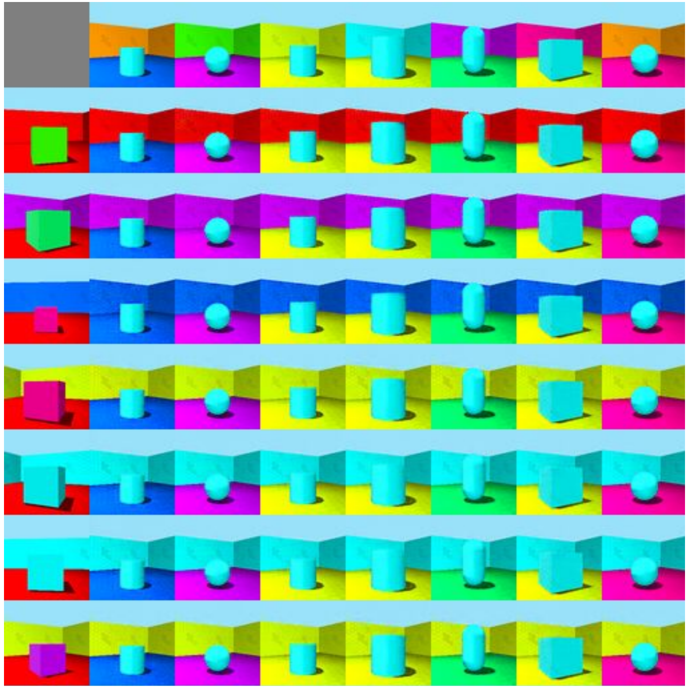
Results

Shapes-3D-B

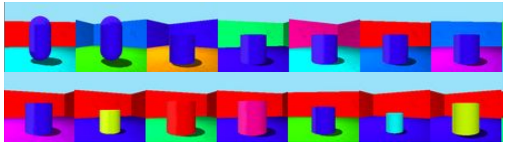


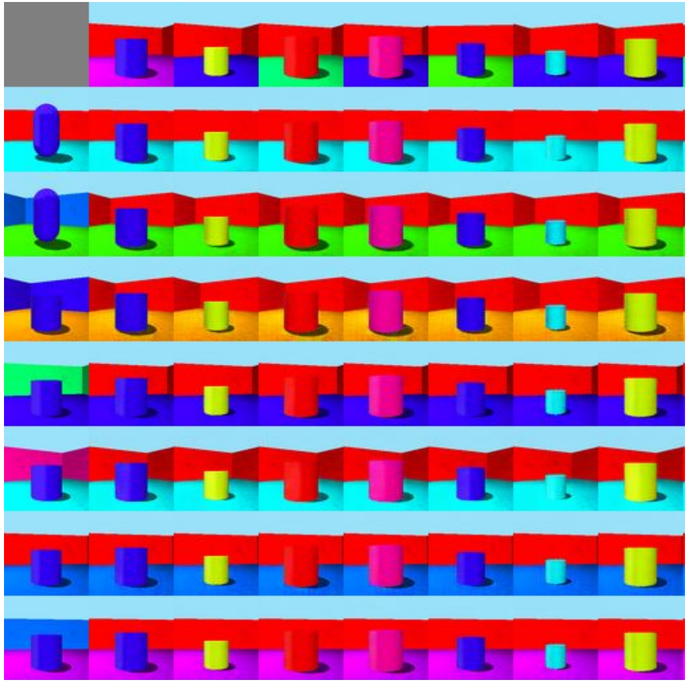
shared: wall color, size
source: object color, orient.
target: shape, floor color

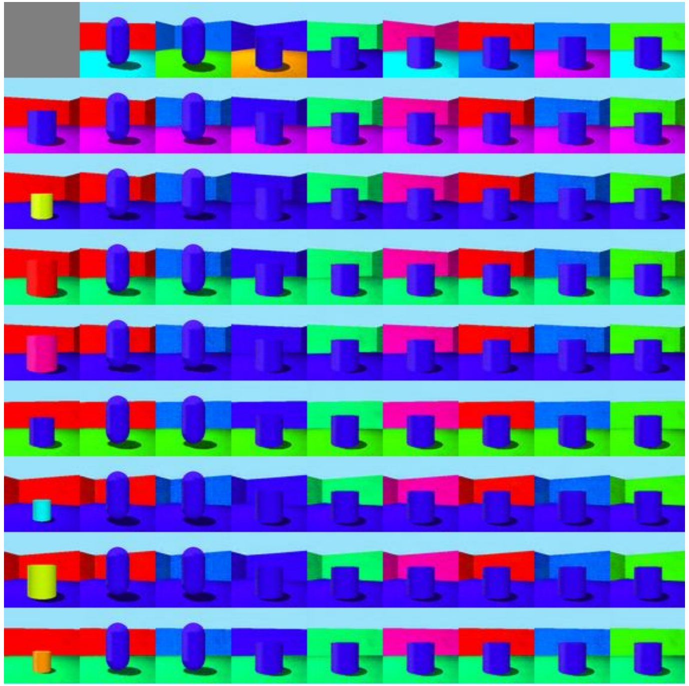




Results


Shapes-3D-C
shared: floor color, orient.
source: wall color, shape
target: size, object color

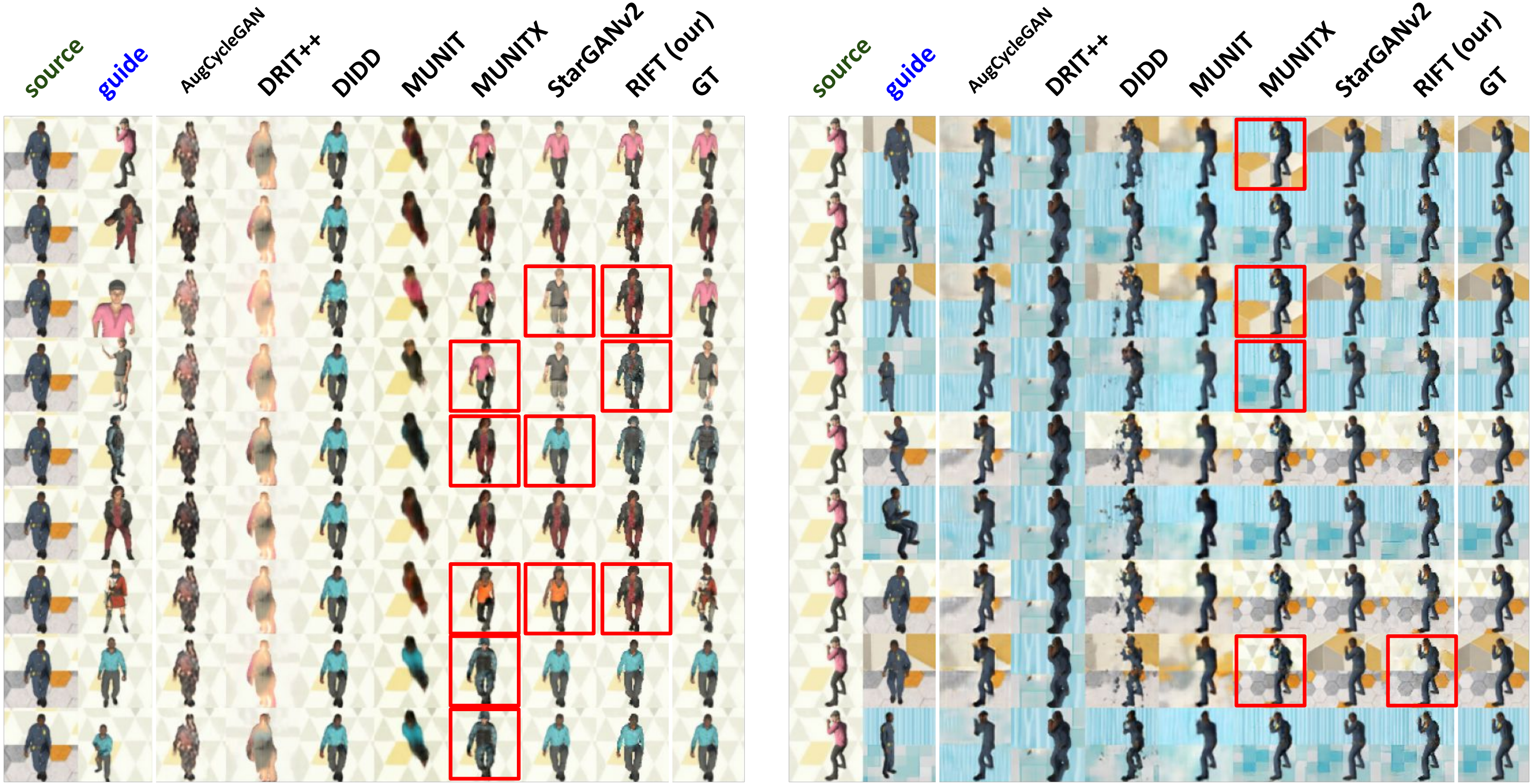

guide (size and object color)
source (floor color and orientation)


guide (shape and wall color)
source (floor color and orientation)

Results

SynAction

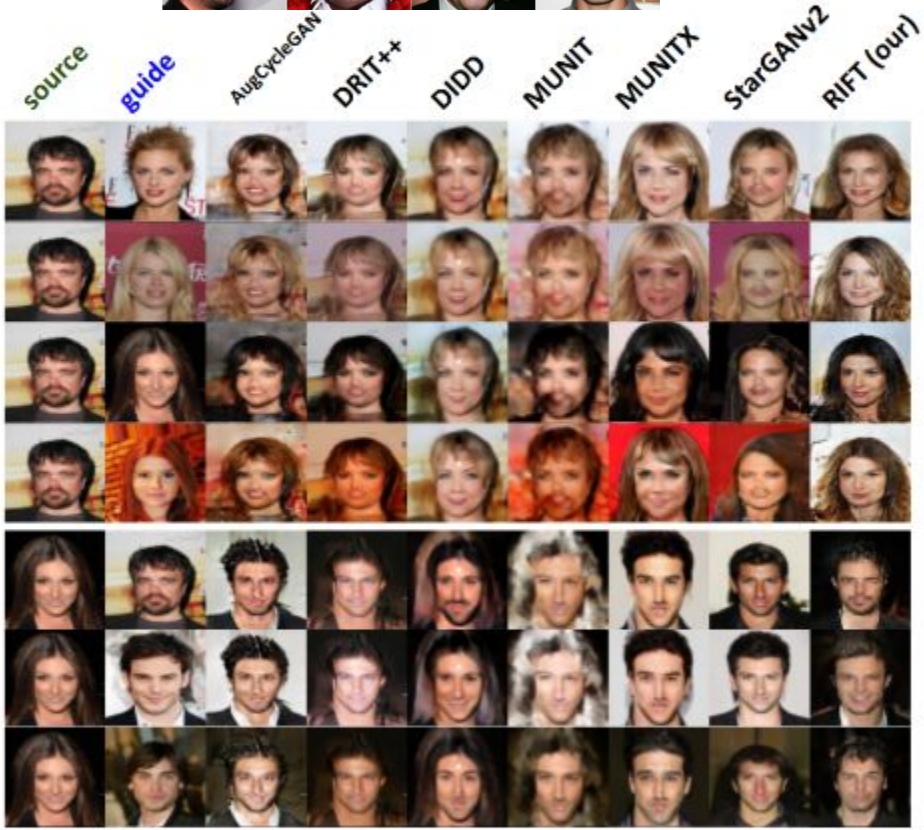
shared: pose
source: background
target: identity/clothing



Results

CelebA

shared: pose, background
source: hair color
target: facial hair



Metrics and Qualitative Results

Manipulation Accuracy (for categorical):

$$\mathrm{ACC}_k^{\mathrm{A}} = p(f_k(F_{\mathrm{A2B}}(a,b)) = y_k^* \mid f_k(a) \neq f_k(b))$$

where the "correct" attribute value equals $y_k^* = f_k(a)$ for shared attributes, and $y_k^* = f_k(b)$ otherwise. For real-

Manipulation Accuracy (for real-valued):

$$\mathrm{ACC}_k^{\mathrm{A}} = p(\|f_k(F_{\mathrm{A2B}}(a,b)) - y_k^*\| \leq \|f_k(F_{\mathrm{A2B}}(a,b)) - y_k'\|)$$

where $y_k^* = f_k(a)$ and $y_k' = f_k(b)$ for shared attributes, and vice-versa otherwise.

Relative Discrepancy (for shapes only):

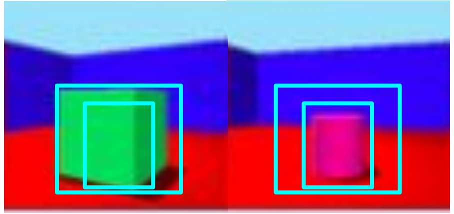
$$\mathrm{RD} = 100 \cdot \frac{\sum_k |\mathrm{ACC}_k^S - \mathrm{ACC}_k^C|}{\sum_k (\mathrm{ACC}_k^S + \mathrm{ACC}_k^C)}.$$

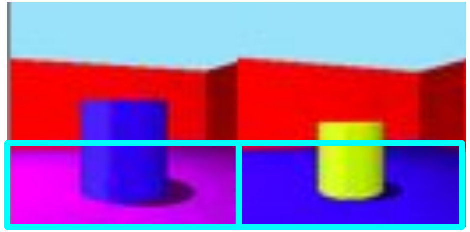
Method	3DS	SA	CA	**AVG**	**RD**
StarGANv2	$\boxed{45}$	$\boxed{\mathbf{82}}$	$\boxed{51}$	<u>59</u>	97
MUNIT	<u>58</u>	37	53	49	56
MUNITX	33	52	55	47	74
DRIT++	18	24	55	32	<u>20</u>
AugCycleGAN	12	37	40	29	<u>20</u>
DIDD	$\boxed{44}$	$\boxed{67}$	$\boxed{\mathbf{64}}$	58	35
RIFT (ours)	**88**	<u>78</u>	<u>60</u>	**75**	**6**
RAND	12	24	49	27	9

Table 1: **Average (AVG↑) manipulation accuracy (ACC) and relative discrepancy (RD↓)** across 3D-Shapes-ABC (3DS), SynAction (SA), and CelebA-FM (CA). Notation: **best**, <u>2nd best</u>.

Remaining Challenges

1. How to deal with attributes that “occupy” very different number of pixels in reconstruction losses (e.g. size vs color)?



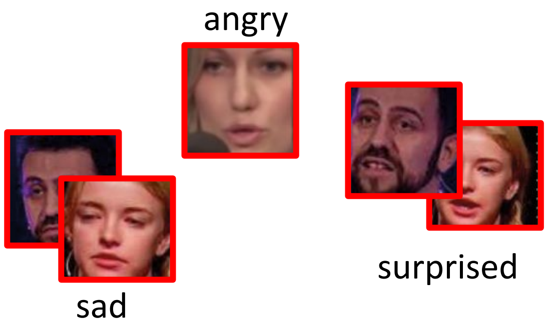


2. What if attributes are varied in both but have different distributions? (e.g. 3% females are blonde, but 50% of males are blonde)

Takeaway

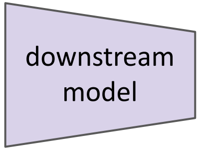
We can use unsupervised alignment to discover domain-specific factors of variability without any supervision!

Applications: **Interpretability and Control**

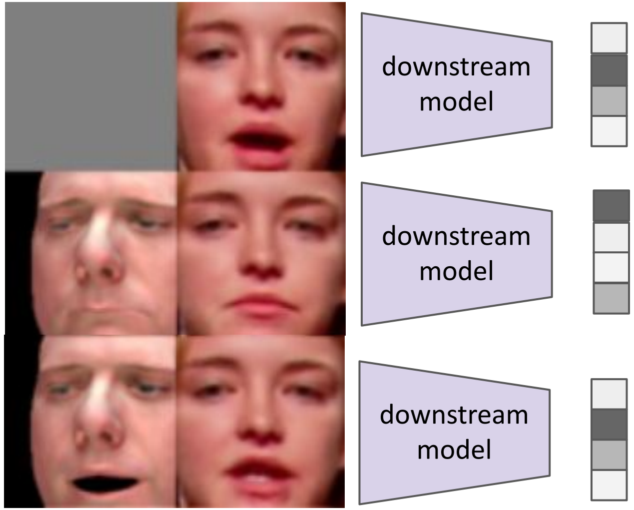


Train PuppetGAN!

I wonder how much my

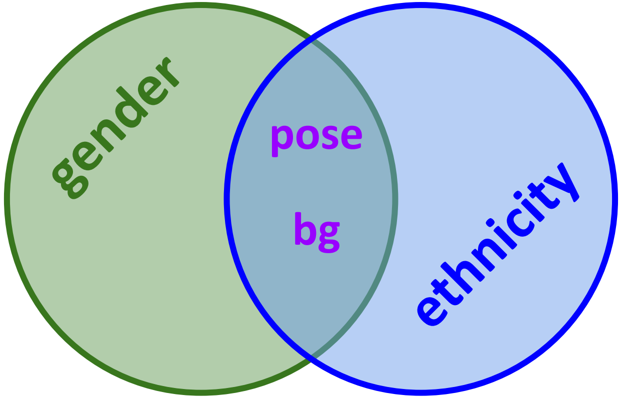


(e.g. emotion recognition model)
is sensitive to **mouth openness**?

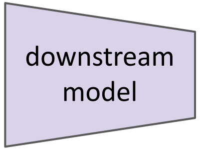


Applications: **Interpretability and Control**

Train factors of variability **Test**



I wonder whether my



is sensitive to factors of variability **absent** in train, but **present** in test?

Train RIFT!

Applications: **Interpretability and Control**

texture

RIFT

LRMF + ?

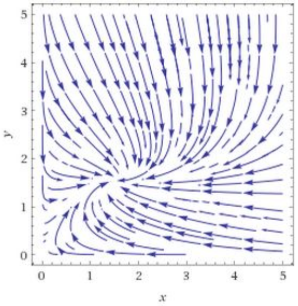
size

location

shape

PuppetGAN

$dx/dt = f(x, t)$



“3DP3: 3D Scene Perception via Probabilistic Programming” by Gothoskar et al.

Learning better one-to-one mappings

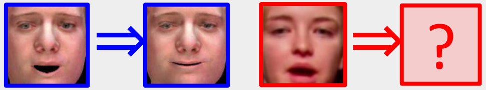
We can get **stable** alignment dy **dualizing** the logistic discriminator! (ICLR-W’18)

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

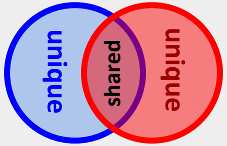
Manipulating factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)



We can manipulate attributes **unique** to each domain independently from those **shared** across domains!

(in submission)



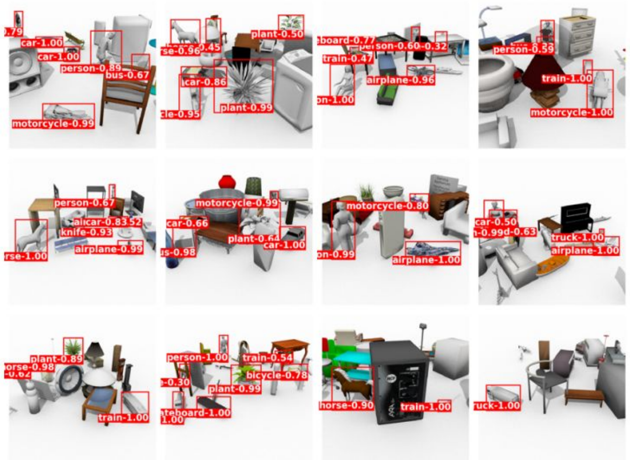
Thank you for your attention!

Questions?

Other research

multi-view RGB → 3D pose
no 3D GT, no camera calibration,
only synchronized RGB + 2D GT for training

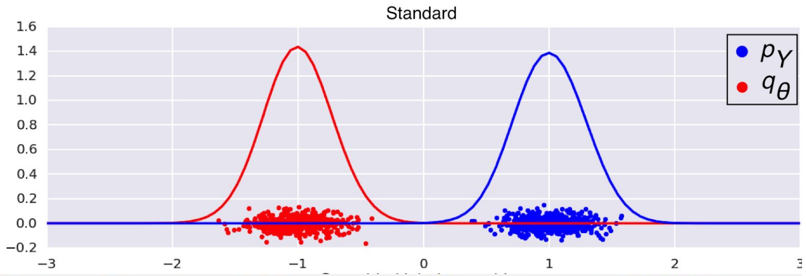
"**MetaPose**: Fast 3D pose from multiple views without 3D supervision",
Usman, Tagliasacchi, Saenko, Sud (CVPR22)

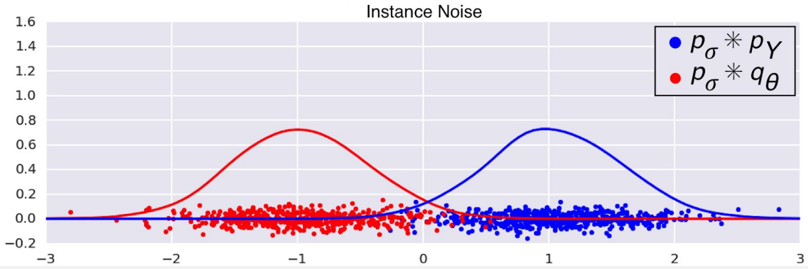


"**Syn2Real**: A New Benchmark for Synthetic-to-Real Visual DA",
Peng, Usman, …, Hoffman, Saenko

Backup deck begins

Instance noise in the discriminator might help. Closed-form regularizer exist.





Regularized Jensen-Shannon GAN

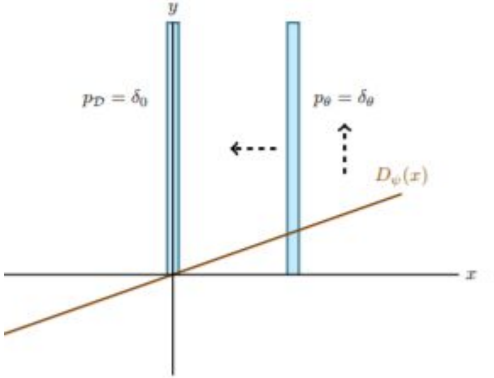
$$F_\gamma(\mathbb{P},\mathbb{Q};\varphi) \;=\; \mathbf{E}_\mathbb{P}\left[\ln(\varphi)\right] + \mathbf{E}_\mathbb{Q}\left[\ln(1-\varphi)\right] - \frac{\gamma}{2}\Omega_{JS}(\mathbb{P},\mathbb{Q};\varphi)$$

$$\Omega_{JS}(\mathbb{P},\mathbb{Q};\varphi) := \mathbf{E}_\mathbb{P}\left[(1-\varphi(\mathbf{x}))^2||\nabla\phi(\mathbf{x})||^2\right] + \mathbf{E}_\mathbb{Q}\left[\varphi(\mathbf{x})^2||\nabla\phi(\mathbf{x})||^2\right]$$

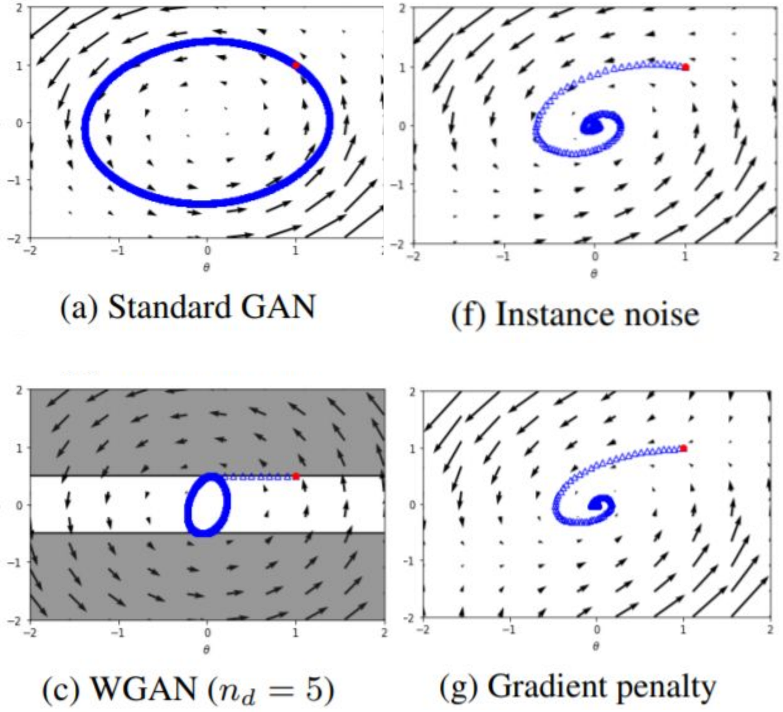
but requires figuring out a good annealing schedule

[**“Stabilizing Training of Generative Adversarial Networks through Regularization”,** Roth et al, NeurIPS’17]
[**“Instance Noise: A trick for stabilising GAN training”**, Ferenc Huszár, inference.vc]

A “toy GAN problem” confirms it.



$$D_\psi(x) = \psi \cdot x$$
$$p_\theta = \delta_\theta$$
$$p_{\mathcal{D}} = \delta_0$$



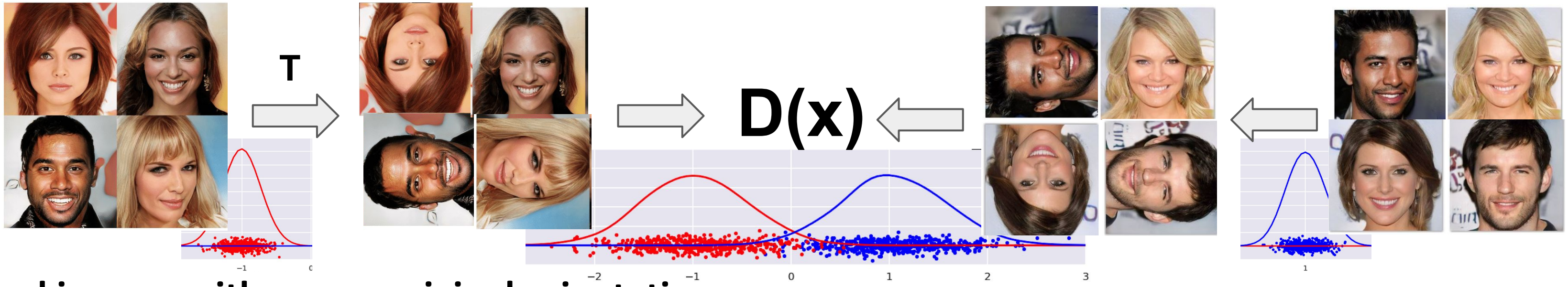
[**“Which Training Methods for GANs do actually Converge?”**, Mescheder et al., ICML’18]

Let’s extend to arbitrary augmentations.

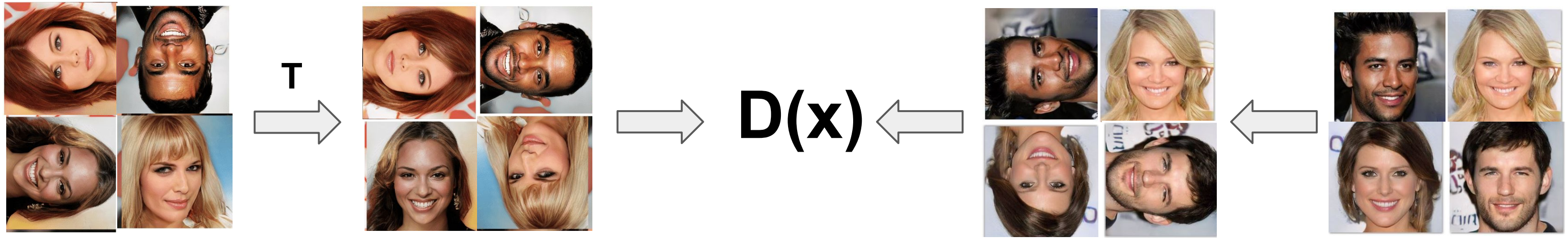
Assume augmentation T(x) randomly flips an image by [0, 90, 180, 270] and we apply T(x) “as instance noise” before passing them to D(x) to make images “less separable”.

“good” generated images

real images



generated images with wrong original orientation



[**“Training Generative Adversarial Networks with Limited Data”**, Karras et al., NeurIPS’20]

Here is what you get - “leaking augmentation”.

T(x) is flip

T(x) is color shift

[**“Training Generative Adversarial Networks with Limited Data”**, Karras et al., NeurIPS’20]

How to avoid “leaking augmentation”?

real images — **generated images**

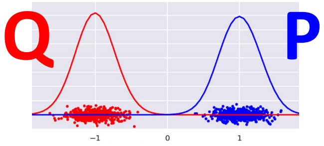
$$\boxed{P} \xrightarrow{\mathbf{T}} \boxed{T(P)} \rightarrow \mathbf{D(x)} \leftarrow \boxed{T(Q)} \xleftarrow{\mathbf{T}} \boxed{Q}$$

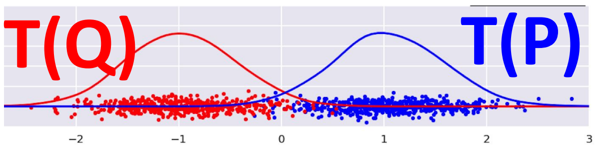
We want T(x) such that **T(P) = T(Q) ⇔ P = Q ,**

i.e. we want an *invertible* operator “T: distribution □ distribution”.

Not same as an invertible augmentation T(x)!

Example: $T(P) = P * Gaussian(0, 1)$, i.e. $T(x) = x + \varepsilon$, $\varepsilon \sim N(0, 1)$.





[**“Training Generative Adversarial Networks with Limited Data”**, Karras et al., NeurIPS’20]

In general, these transformations (rotation, shift, etc.) induce operators over the space of distributions and have some group structure.

(In appendix) they show sufficient conditions for spectra of these linear operators not containing zeros ⇒ operators themselves being invertible.

[**“Training Generative Adversarial Networks with Limited Data”**, Karras et al., NeurIPS’20]

Teaser: core results

Learning better one-to-one mappings

We can get **stable** alignment wrt **powerful** discriminator families using normalizing flows! (NeurIPS20)

Defending models against performing adversarial attacks **on themselves** improves **semantic consistency**! (NeurIPS19)

Manipulating individual factors with cross-domain supervision

We can alter a **single specific attribute** of real images using **only synthetic supervision**! (ICCV19 Oral)

We can infer which attributes are **unique** to each domain and **modulate** them in a **controlled** manner! (in submission)

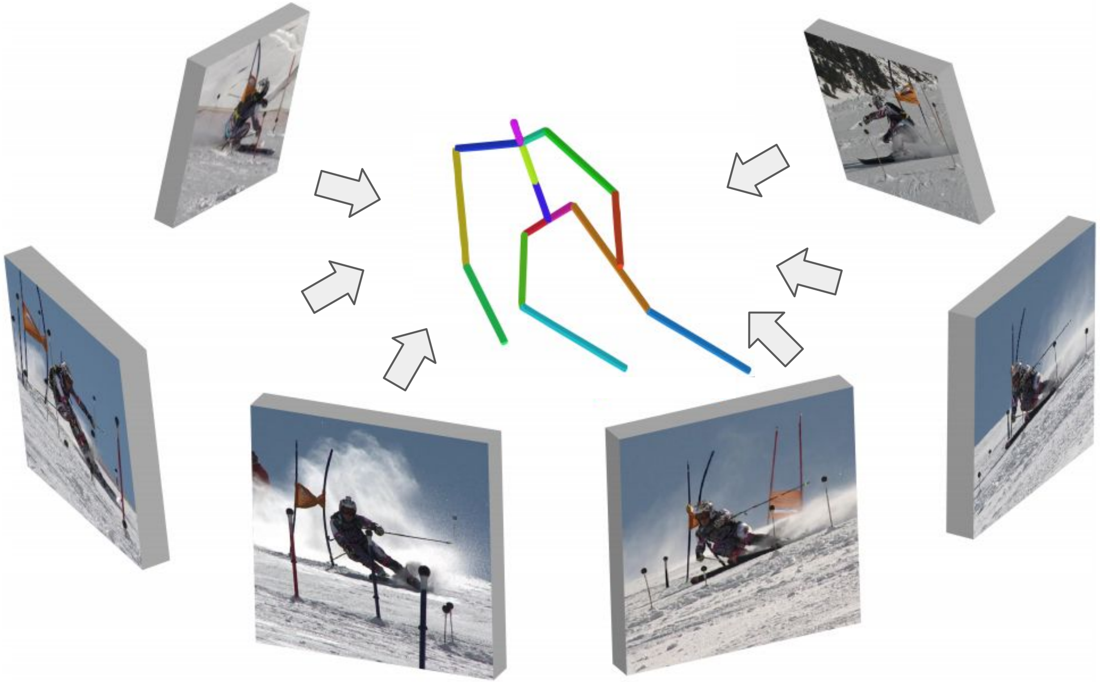
Bonus: Multi-view / 3D simulation

Neural networks can be trained to perform **regularized bundle adjustment** to robustly estimate 3D poses from uncalibrated multi-view RGB **without 3D supervision!** (CVPR22)

We generated one of current de-facto standard datasets for synthetic-to-real adaptation (Syn2Real)

Task

We have synchronized **multi-view** RGB footage
and we want to estimate **3D human pose** from it.



✗ no camera calibration

✗ no GT 3D poses

some 2D pose annotations

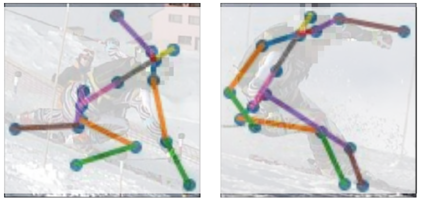
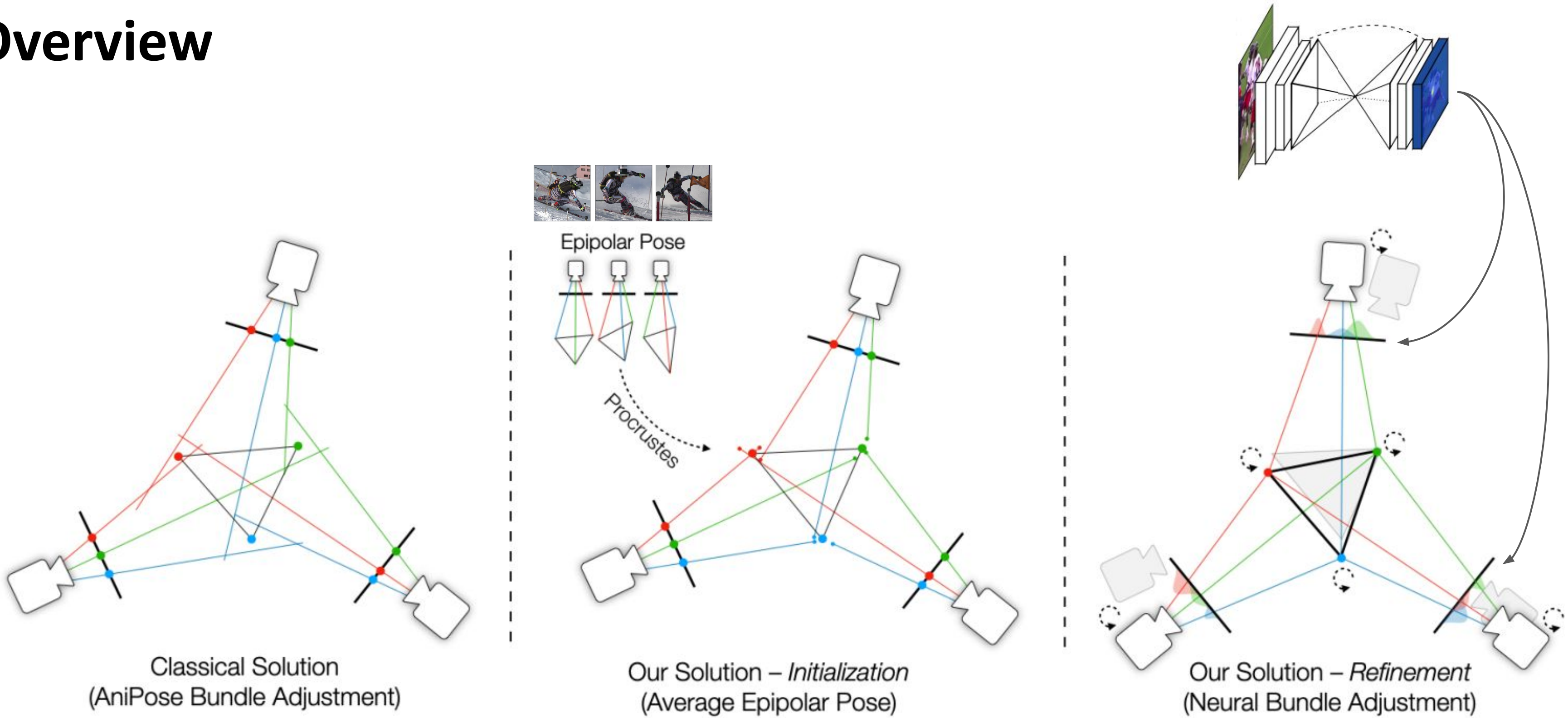
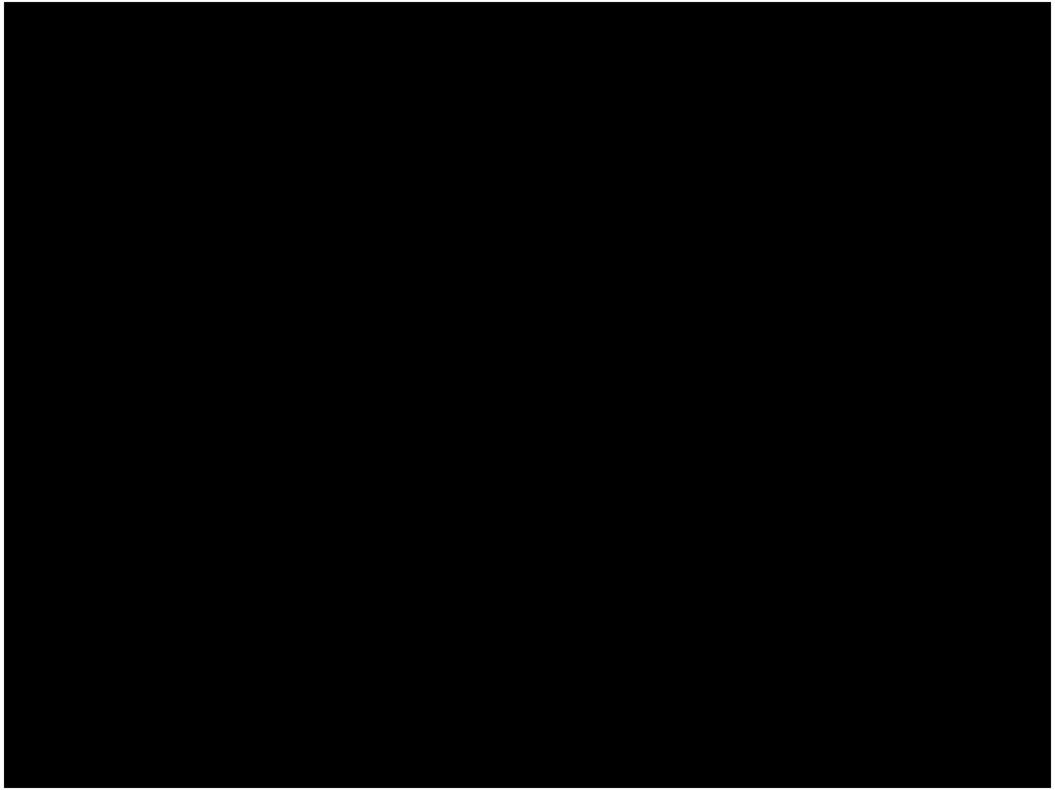


image from “Learning Monocular 3D Human Pose Estimation from Multi-view Images”, Rhodin et al (CVPR18)

Overview



"MetaPose: Fast 3D pose from multiple views without 3D supervision", Usman, Tagliasacchi, Saenko, Sud (CVPR22)



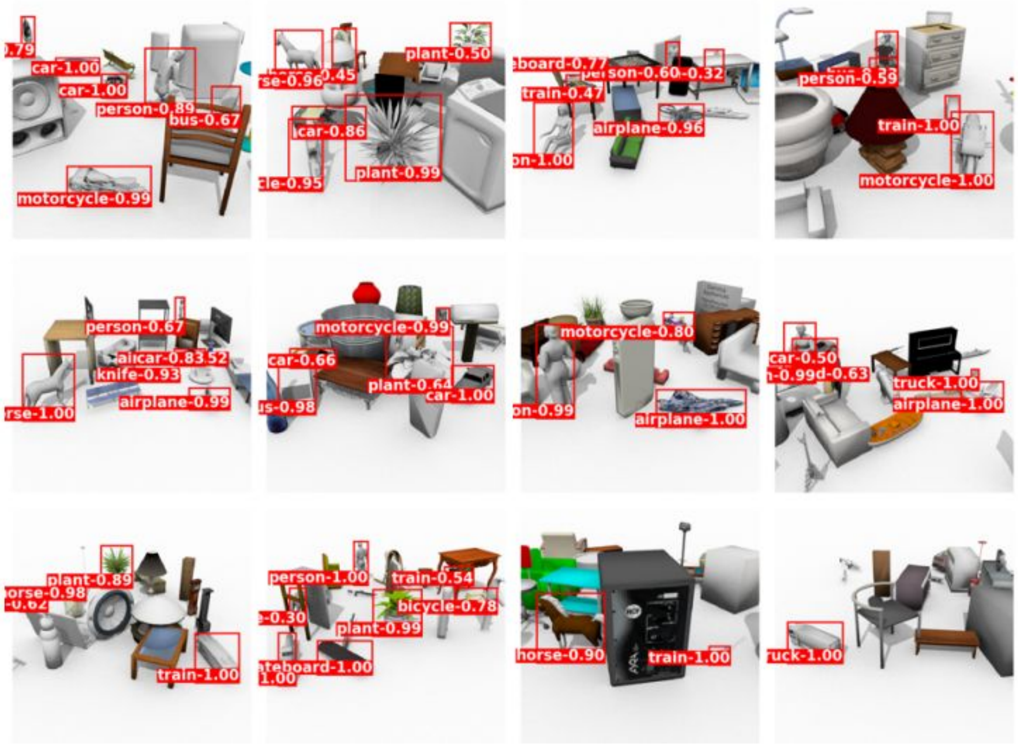
Human3.6M

Method	PMPJPE↓ 4	PMPJPE↓ 2	NMPJPE↓ 4	NMPJPE↓ 2	Δt [s]
Isakov et al. [19]	20	-	-	-	-
AniPose [25] w/ GT	75	167	103	230	7.0
Rhodin et al. [37]	65	-	80	-	-
CanonPose [44]	53	-	82	-	-
EpipolarPose (EP) [27]	71	-	78	-	-
Iqbal et al. [18]	55	-	66	-	-
MetaPose (S1)	74	87	83	95	**0.2**
MetaPose (S1+S2)	**32**	**44**	**49**	**55**	0.3

SkiPose

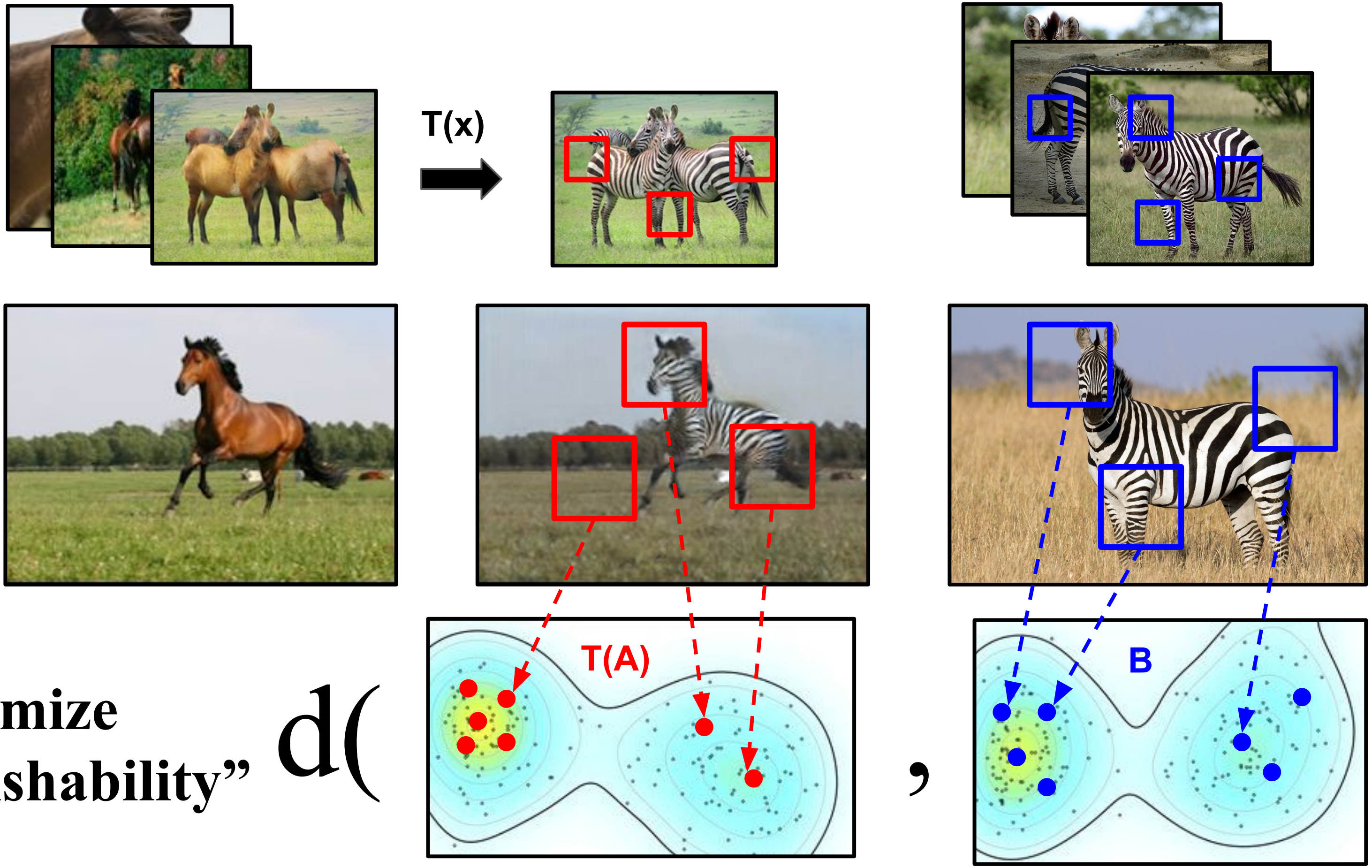
Method	PMPJPE↓ 6	PMPJPE↓ 2	NMPJPE↓ 6	NMPJPE↓ 2	Δt [s]
AniPose [25] w/ GT	50	62	221	273	7.0
Rhodin et al. [37]	-	-	85	-	-
CanonPose (CP) [44]	90	-	128	-	-
MetaPose (S1)	81	86	140	144	**0.3**
MetaPose (S1+S2)	**42**	**50**	**53**	**59**	0.4

"MetaPose: Fast 3D pose from multiple views without 3D supervision", Usman, Tagliasacchi, Saenko, Sud (CVPR22)



“Syn2Real: A New Benchmark for Synthetic-to-Real Visual Domain Adaptation”, Peng, Usman, …, Hoffman, Saenko

Solution: Image Translation / Domain Alignment



minimize “distinguishability” d(T(A) , B)

[I have all the other work]

[downstream model]

takeaway