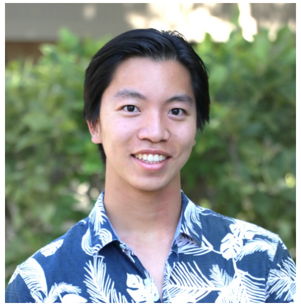
Connect, Not Collapse: Explaining Contrastive Learning for Unsupervised Domain Adaptation


Kendrick Shen*


Robbie Jones*


Ananya Kumar*


Sang Michael Xie*

Jeff Z. HaoChen


Tengyu Ma


Percy Liang

Unsupervised domain adaptation (UDA)

Labeled source domain

Clock

Unsupervised domain adaptation (UDA)

Labeled source domain

Clock

Unlabeled target domain

?

Unsupervised domain adaptation (UDA)

Labeled source domain	Unlabeled target domain
	
Clock	*?*

Goal: high accuracy on target domain (without labels)

Classical approach for UDA

Labeled source domain

Source
representations

Unlabeled target domain

Target
representations

Classical approach for UDA

Labeled source domain

Source representations

High accuracy (given labels)

Unlabeled target domain

Target representations

Classical approach for UDA

Labeled source domain

Unlabeled target domain

Source representations

Target representations

High accuracy (given labels)

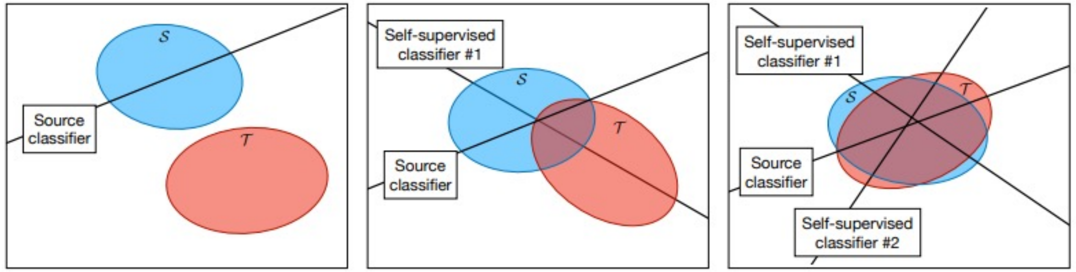
Match distributions

Classical approach for UDA

Motivated by theories such as $H\Delta H$ divergence (Ben-David et al 2010): want source and target reps to be “indistinguishable” to get good target accuracy

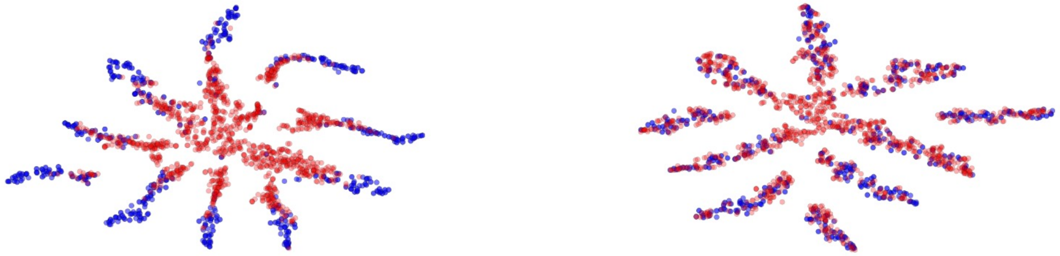
Classical approach for UDA

Motivated by theories such as $H\Delta H$ divergence (Ben-David et al 2010): want source and target reps to be “indistinguishable” to get good target accuracy



(a) Source classifier only (b) adding a self-supervised task (c) adding more tasks

UDA-SS (Sun et al. 2019)



(a) Non-adapted (b) Adapted

DANN (Ganin et al. 2016)

Pre-training for UDA

Step 1: pre-train on unlabeled data (combined source + target)

Pre-training for UDA

Step 1: pre-train on unlabeled data (combined source + target)

Step 2: fine-tune on labeled data (source)

Pre-training for UDA

Step 1: pre-train on unlabeled data (combined source + target)

Step 2: fine-tune on labeled data (source)

Step 3: evaluate accuracy (target)

Pre-training for UDA

- Unsupervised representation learning for UDA inspired by e.g., Blitzer et al. 2007

Pre-training for UDA

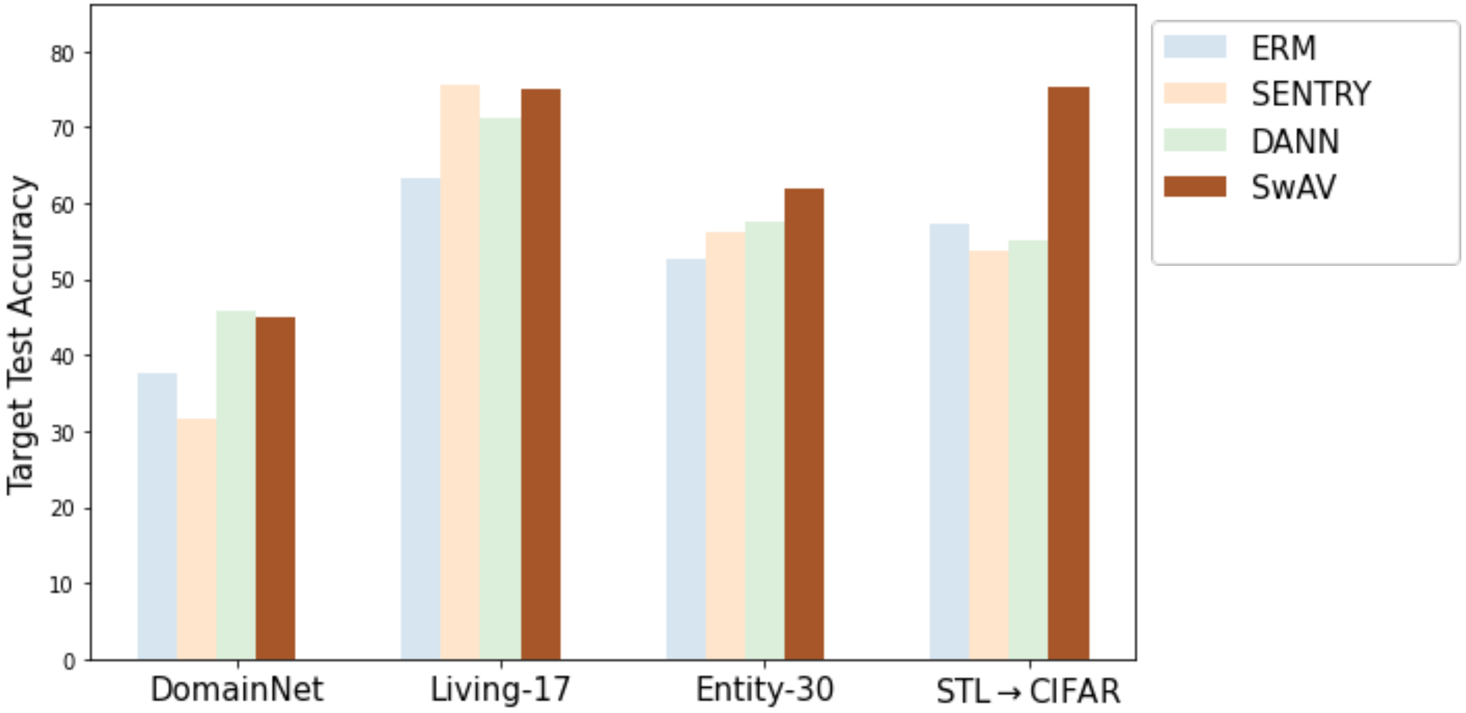
- Unsupervised representation learning for UDA inspired by e.g., Blitzer et al. 2007
  - Representations learned on diverse unlabeled data may relate domains, enabling transfer. But not typically used for deep models

Pre-training for UDA

- Unsupervised representation learning for UDA inspired by e.g., Blitzer et al. 2007
  - Representations learned on diverse unlabeled data may relate domains, enabling transfer. But not typically used for deep models
- What about modern pre-training methods, such as contrastive learning (van den Oord et al. 2018, Chen et al. 2020, Caron et al. 2020)?

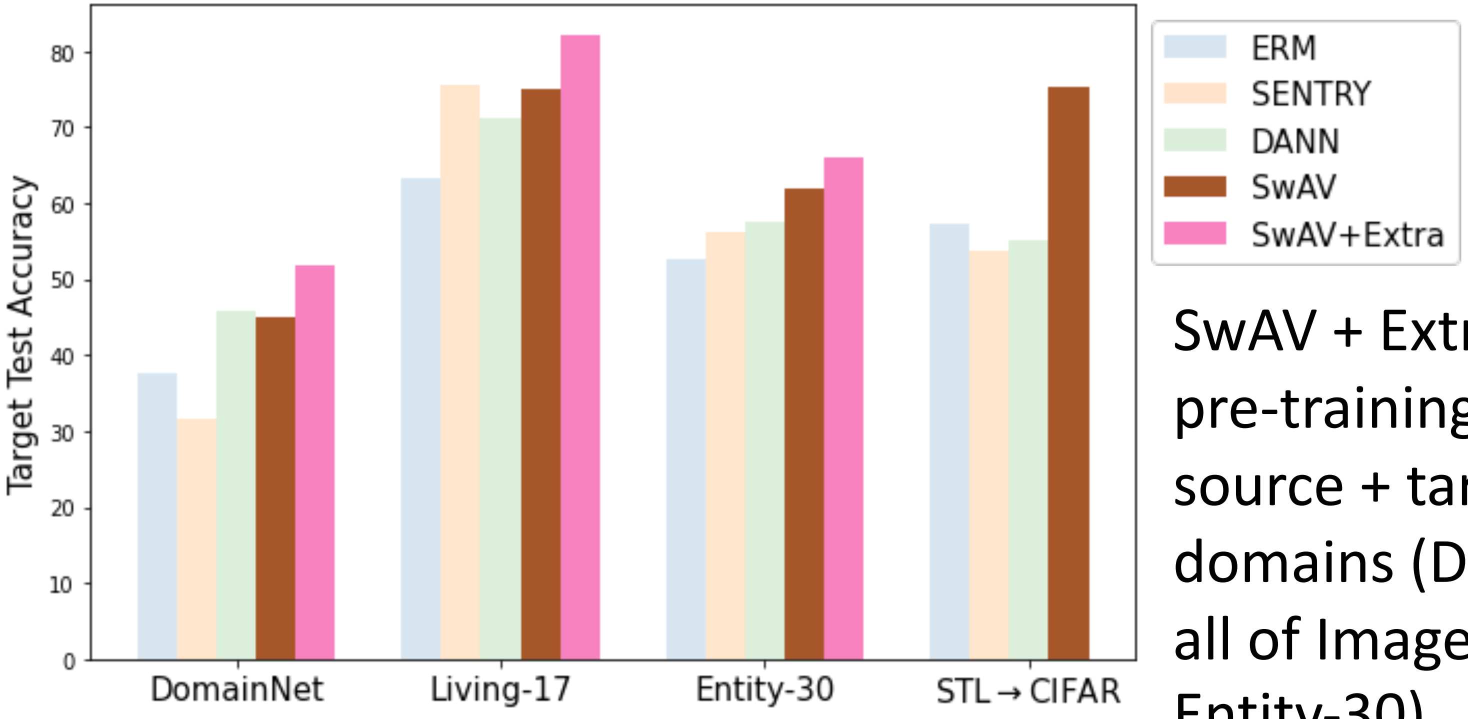
Contrastive pre-training for UDA

Contrastive pre-training (SwAV, Caron et al. 2020) is competitive with UDA methods (even when all methods use the same augmentations)



Contrastive pre-training for UDA

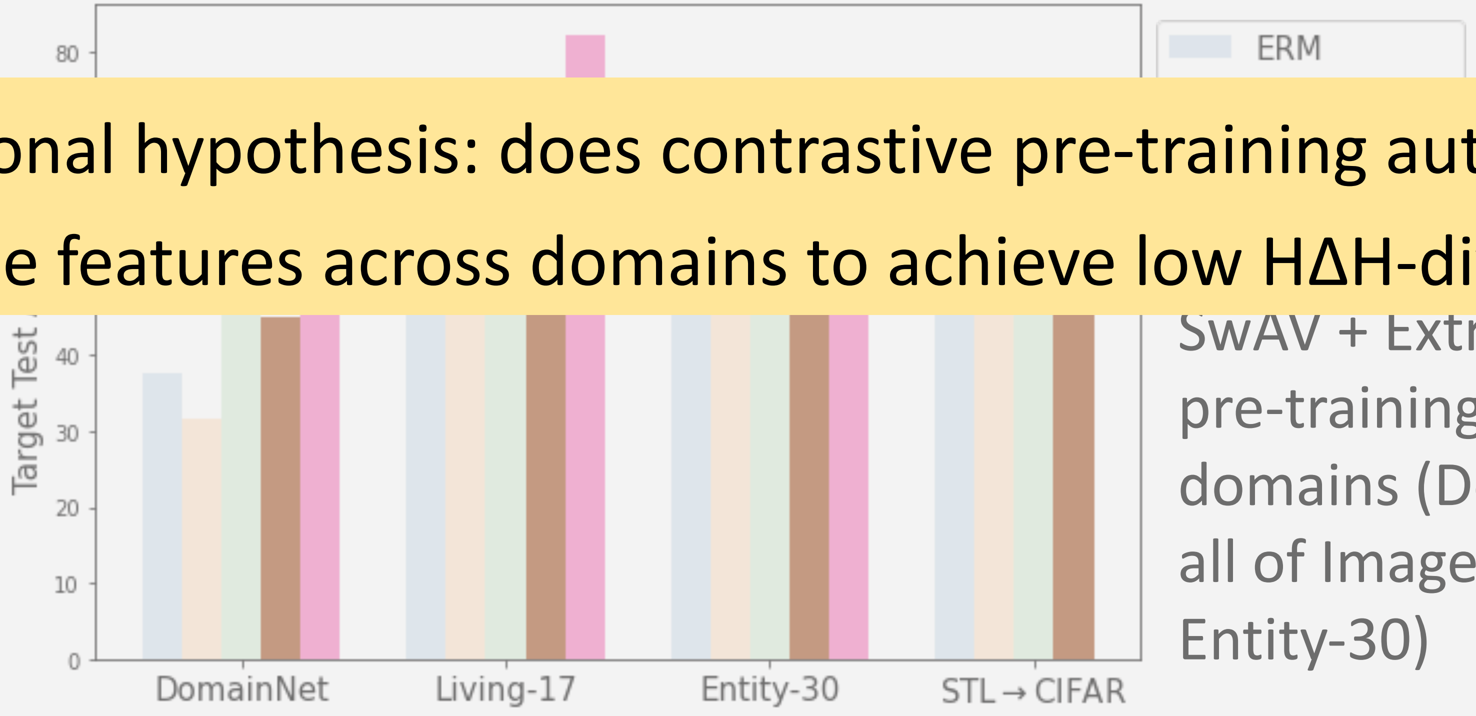
Contrastive pre-training (SwAV, Caron et al. 2020) is competitive with UDA methods (even when all methods use the same augmentations)



SwAV + Extra: unlabeled pre-training data beyond source + target = all 4 domains (DomainNet) or all of ImageNet (Living-17, Entity-30)

Contrastive pre-training for UDA

Contrastive pre-training (SwAV, Caron et al. 2020) is competitive with UDA methods (even when all methods use the same augmentations)



Conventional hypothesis: does contrastive pre-training automatically merge the features across domains to achieve low HΔH-divergence?

SwAV + Extra: unlabeled pre-training data = all 4 domains (DomainNet) or all of ImageNet (Living-17, Entity-30)

Contrastive pre-training doesn't bring domains together

Inspect DANN vs contrastive learning features: train discriminator between domains or between classes

Domain 1 (Sketch)

Domain 2 (Real)

Class 1 (Butterfly)

Class 2 (Clock)

Contrastive pre-training doesn’t bring domains together

Inspect DANN vs contrastive learning features: train discriminator between domains or between classes

Domain 1 (Sketch)

Domain 2 (Real)

Class 1
(Butterfly)

Between classes
DANN: 6% err
Contrastive: 7% err

Class 2
(Clock)

Between domains
DANN: 14% err
Contrastive: 8% err

Contrastive pre-training doesn't bring domains together

Inspect DANN vs contrastive learning features: train discriminator between domains or between classes

Domain 1 (Sketch)

Domain 2 (Real)

Class 1
(Butterfly)

Class 2
(Clock)

Between domains
DANN: 14% err
Contrastive: 8% err

Between classes
DANN: 6% err
Contrastive: 7% err

Pre-training does not produce domain invariant features,
and domains are about as "far apart" as classes!

Contrastive pre-training for UDA

- Performs competitively with strong baselines: SENTRY (Prabhu et al. 2021), DIRT-T (Shu et al. 2018), and DANN (Ganin et al. 2016)

Contrastive pre-training for UDA

- Performs competitively with strong baselines: SENTRY (Prabhu et al. 2021), DIRT-T (Shu et al. 2018), and DANN (Ganin et al. 2016)
- Instead of collapsing domains together, learns features that vary substantially across domains

Contrastive pre-training for UDA

- Performs competitively with strong baselines: SENTRY (Prabhu et al. 2021), DIRT-T (Shu et al. 2018), and DANN (Ganin et al. 2016)
- Instead of collapsing domains together, learns features that vary substantially across domains

Why do these features still generalize to the target without domain invariance?

Outline

- Setup: augmentation graph
- Intuitions and theoretical results
  - Main intuitions (toy example)
  - Results for stochastic block model & beyond
  - Contrastive pre-training vs. ERM & DANN
- Test theoretical predictions on real data

Outline

- Setup: augmentation graph
- Intuitions and theoretical results
  - Main intuitions (toy example)
  - Results for stochastic block model & beyond
  - Contrastive pre-training vs. ERM & DANN
- Test theoretical predictions on real data

Setup: augmentation graph

- Contrastive learning hinges on *positive pairs* (augmentations of the same original input)

Setup: augmentation graph

- Contrastive learning hinges on *positive pairs* (augmentations of the same original input)
- Contrastive objective:
  - map positive pairs to similar features

Setup: augmentation graph

- Contrastive learning hinges on *positive pairs* (augmentations of the same original input)
- Contrastive objective:
  - map positive pairs to similar features
  - map augmentations of different inputs to different features

Setup: augmentation graph

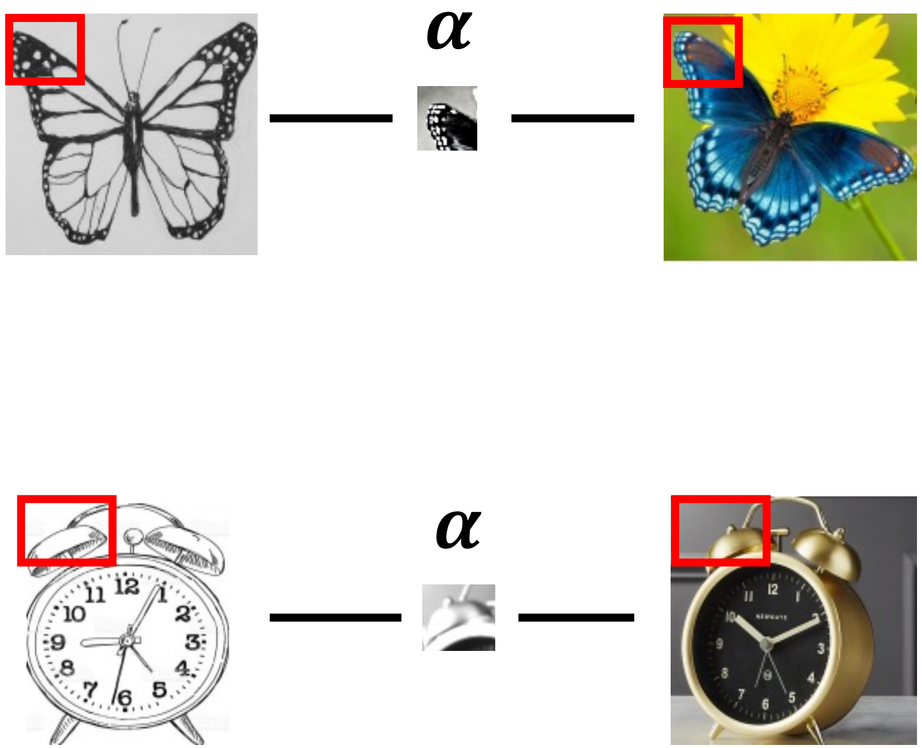
	Domain 1 (Sketch)	Domain 2 (Real)
Class 1 (Butterfly)		
Class 2 (Clock)		

Setup: augmentation graph

Domain 1 (Sketch) Domain 2 (Real)

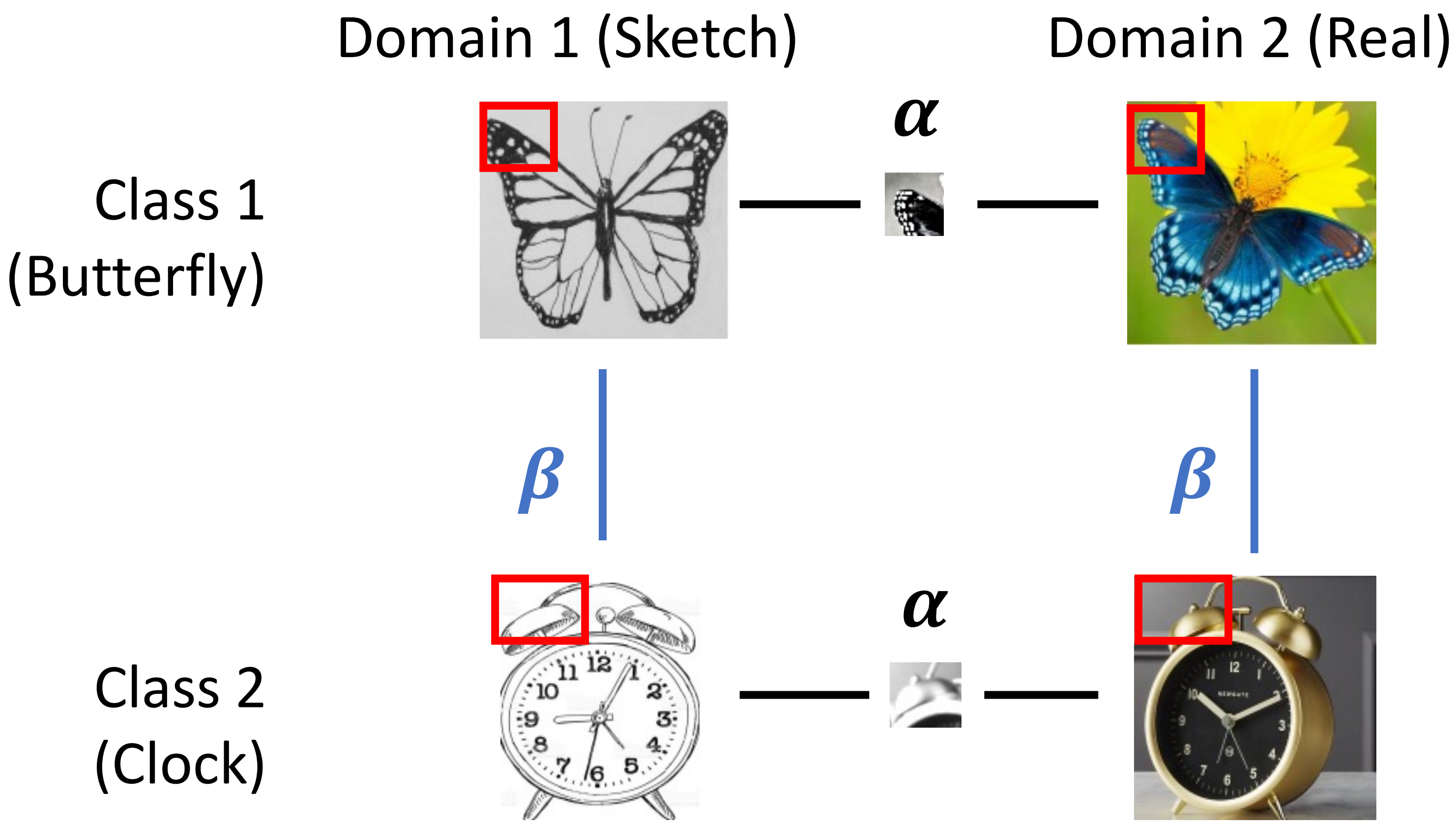
Class 1
(Butterfly)

Class 2
(Clock)

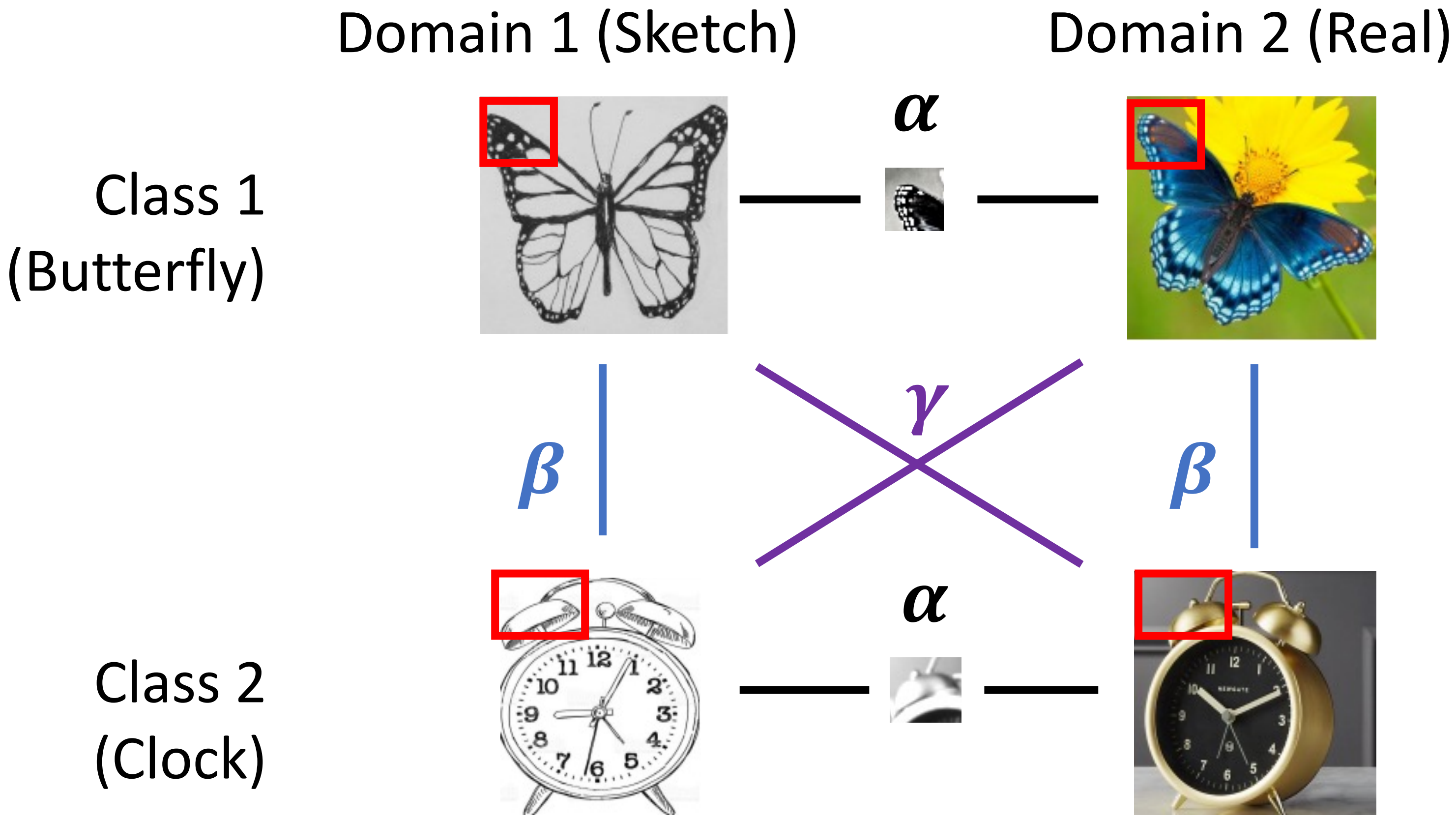


α: probability that augmentations of images coincide

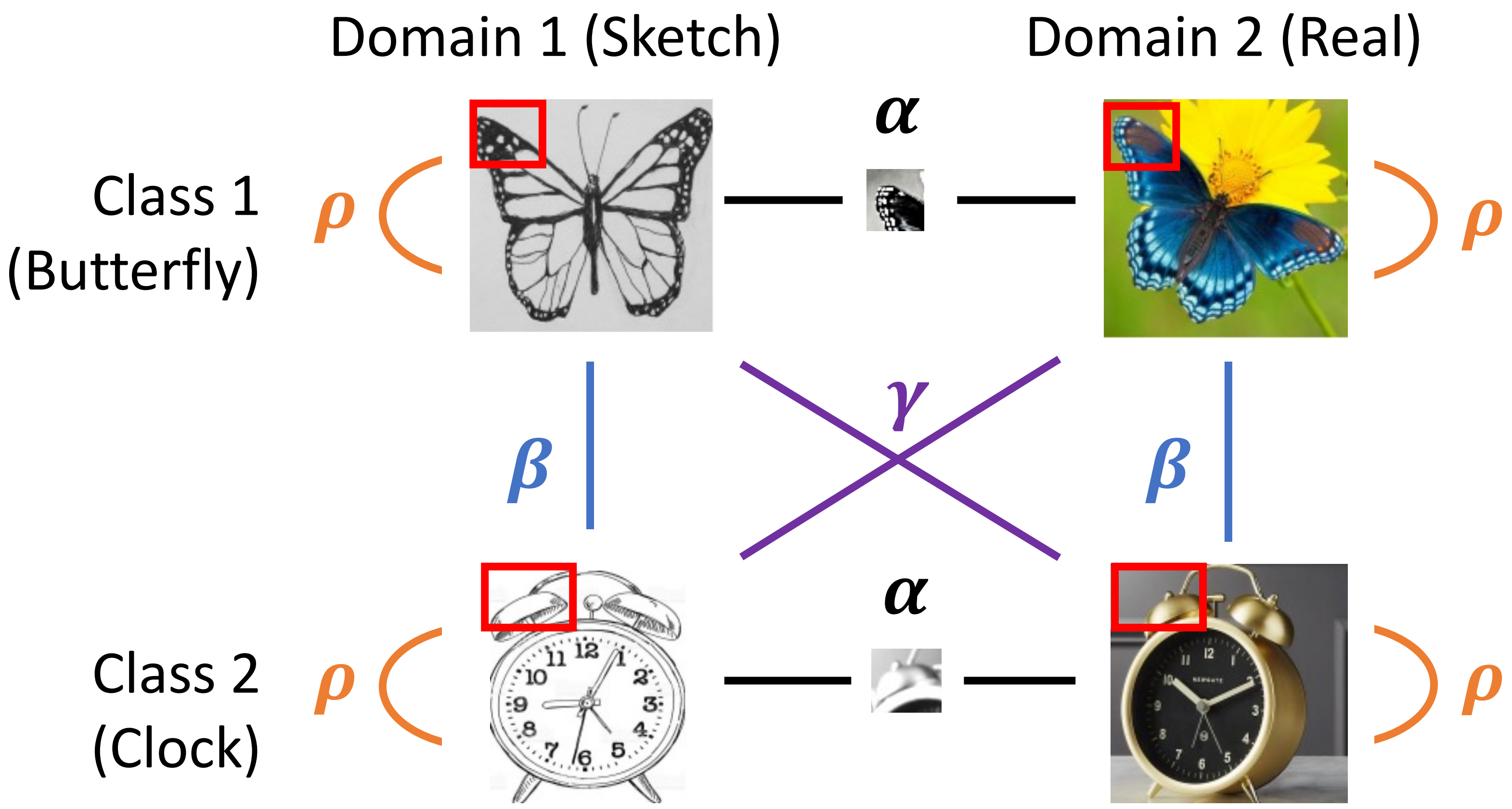
Setup: augmentation graph



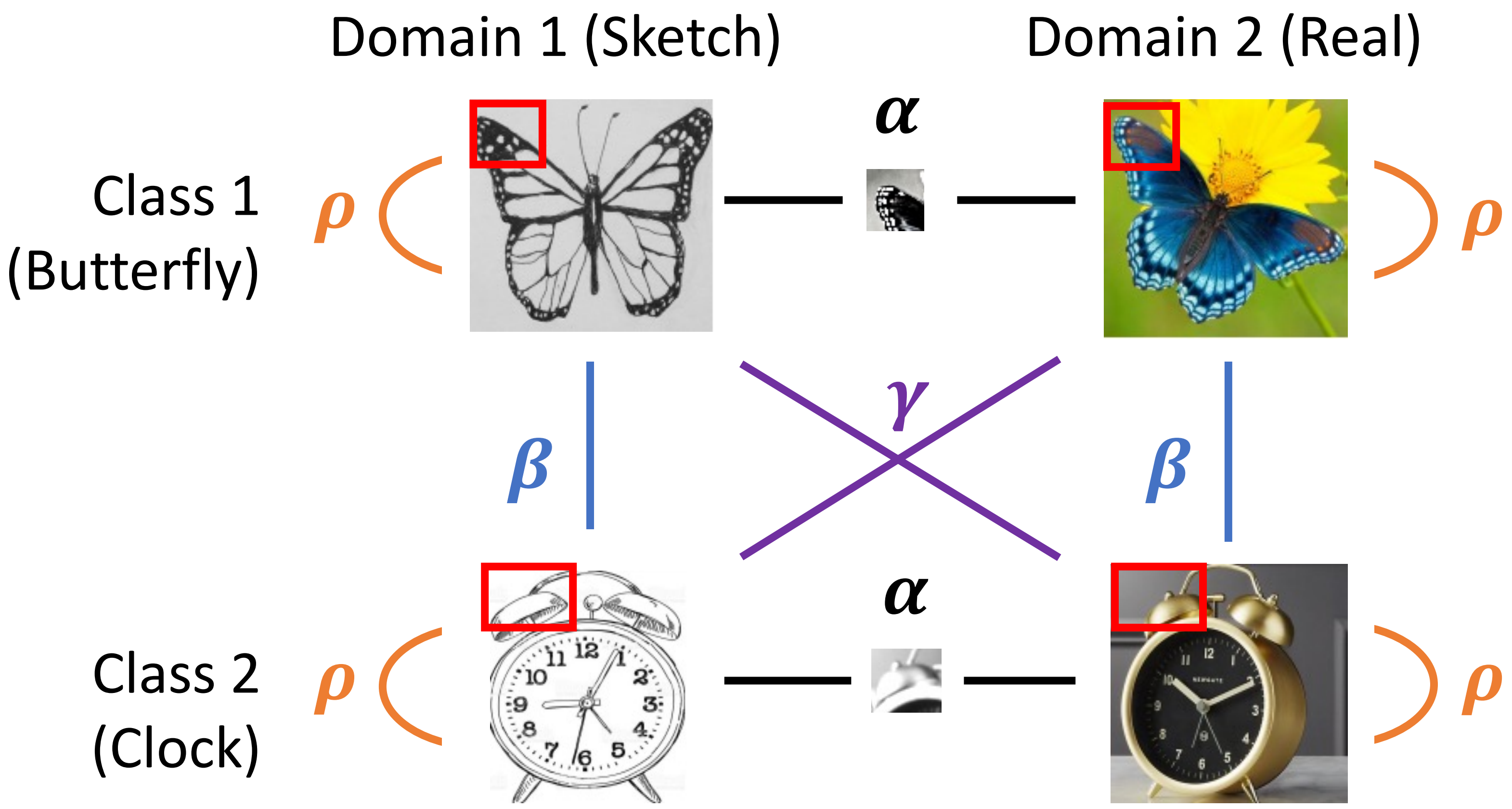
Setup: augmentation graph



Setup: augmentation graph

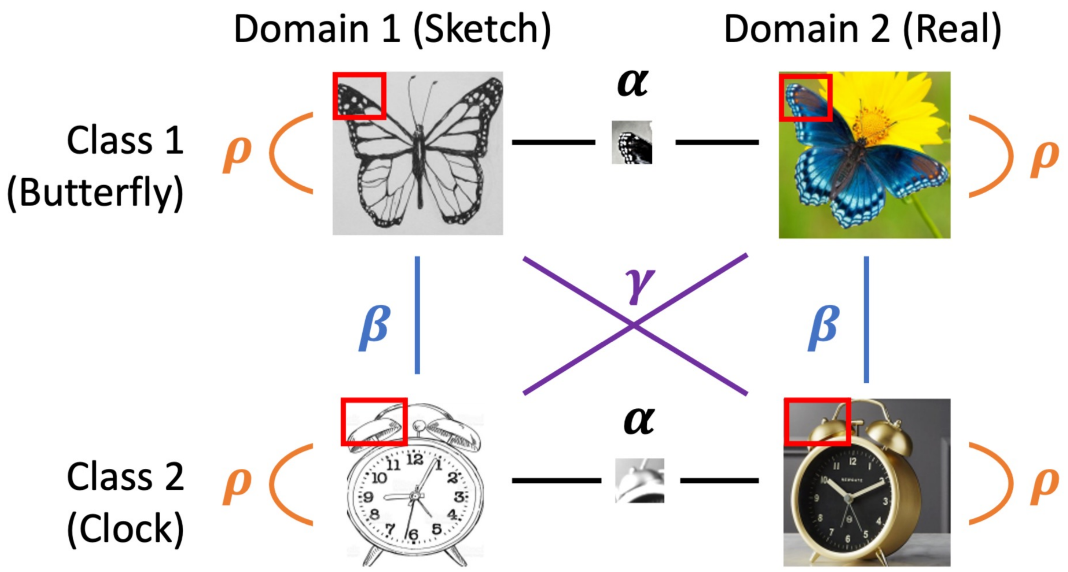


Setup: augmentation graph



Magnitudes of connectivity parameters ρ, α, β, and $\gamma \approx$ similarity of augmentations

Setup: augmentation graph



	Sketch clock	Sketch butterfly	Real clock	Real butterfly
Sketch clock	ρ	β	α	γ
Sketch butterfly	β	ρ	γ	α
Real clock	α	γ	ρ	β
Real butterfly	γ	α	β	ρ

Can express augmentation graph using adjacency matrix A

Outline

- Setup: augmentation graph
- Intuitions and theoretical results
    - Main intuitions (toy example)
    - Results for stochastic block model & beyond
    - Contrastive pre-training vs. ERM & DANN
- Test theoretical predictions on real data

Intuitions & toy example

- Let S_+ be the positive pair distribution, P_U the unlabeled data distribution, and f the feature extractor

Intuitions & toy example

- Let S_+ be the positive pair distribution, P_U the unlabeled data distribution, and f the feature extractor
- We learn $\widehat{f} : X \to R^k$ that minimizes the spectral contrastive loss (HaoChen et al. 2021):

$$\mathcal{L}_{\text{pretrain}}(f) = -2 \cdot \mathbb{E}_{(x,x^+)\sim S_+}\left[f(x)^\top f(x^+)\right] + \mathbb{E}_{x,x'\sim P_U}\left[\left(f(x)^\top f(x')\right)^2\right]$$

Intuitions & toy example

- Let S_+ be the positive pair distribution, P_U the unlabeled data distribution, and f the feature extractor
- We learn $\widehat{f} : X \rightarrow R^k$ that minimizes the spectral contrastive loss (HaoChen et al. 2021):

$$\mathcal{L}_{\text{pretrain}}(f) = -2 \cdot \mathbb{E}_{(x,x^+)\sim S_+}\left[f(x)^\top f(x^+)\right] + \mathbb{E}_{x,x'\sim P_U}\left[\left(f(x)^\top f(x')\right)^2\right]$$

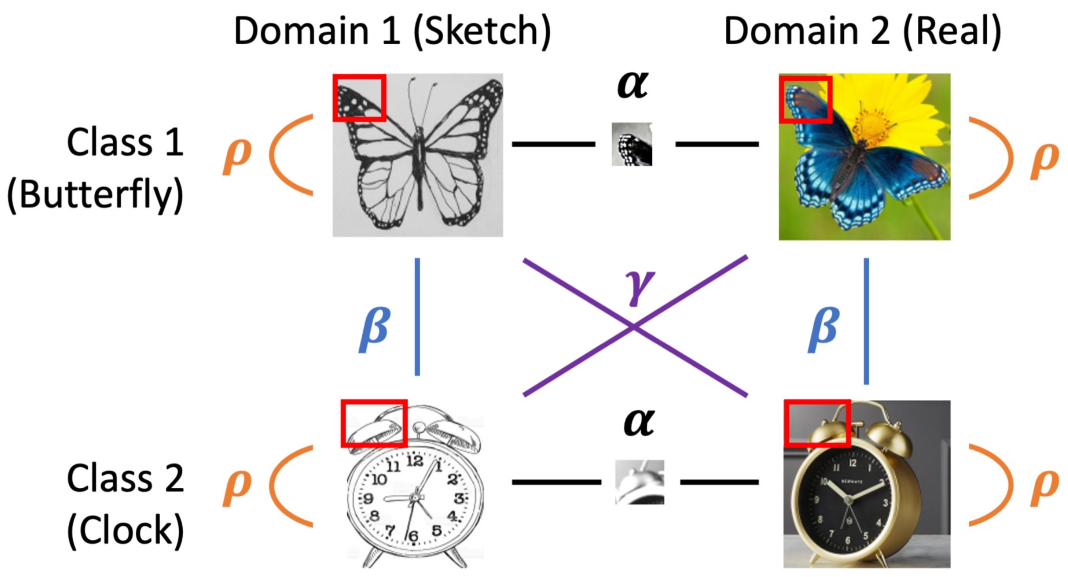
- In our toy example, we can compute $\widehat{f}$ exactly and then visualize the (learned) representations

Intuitions & toy example

- Binary classification, 1 example per class and domain (4 examples total)

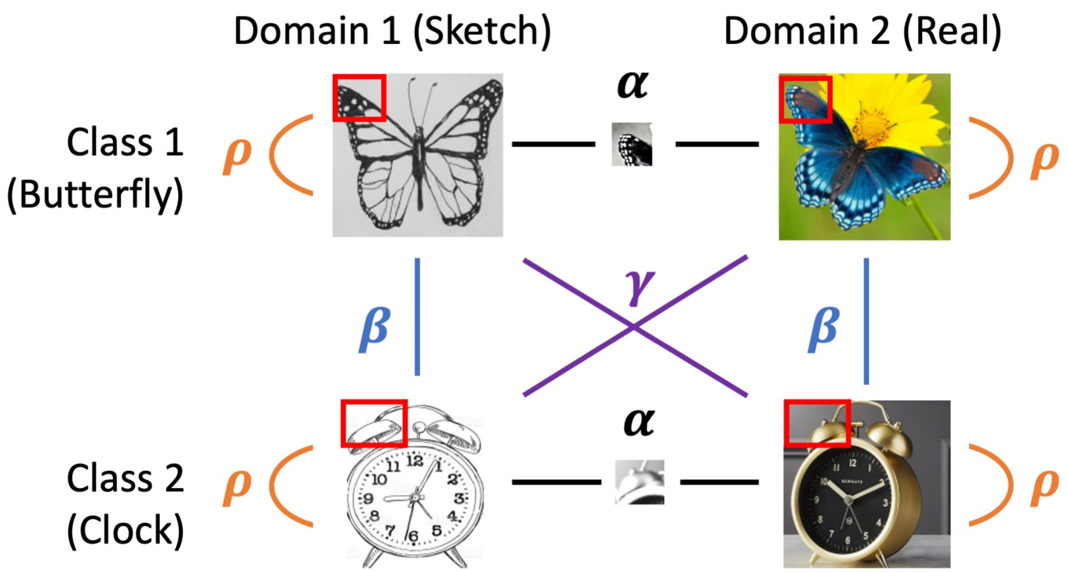
Intuitions & toy example

- Binary classification, 1 example per class and domain (4 examples total)
- Let $\hat{F}: R^{4\times 3}$ be a matrix whose rows contain learned features



Intuitions & toy example

- Binary classification, 1 example per class and domain (4 examples total)
- Let $\hat{F}: R^{4\times 3}$ be a matrix whose rows contain learned features



$$\hat{F} = \begin{bmatrix} \longleftarrow & \hat{f}\,(\text{sketch clock}) & \longrightarrow \\ \longleftarrow & \hat{f}\,(\text{sketch butterfly}) & \longrightarrow \\ \longleftarrow & \hat{f}\,(\text{real clock}) & \longrightarrow \\ \longleftarrow & \hat{f}\,(\text{real butterfly}) & \longrightarrow \end{bmatrix}$$

Intuitions & toy example

- Recall: A is adjacency matrix of positive pair probabilities

Intuitions & toy example

- Recall: A is adjacency matrix of positive pair probabilities
- $\hat{F} = \mathrm{argmin}_{\hat{F}} || A - \hat{F}\hat{F}^T ||_F$ if we use spectral contrastive loss (HaoChen et al. 2021)
- Columns of $\hat{F}$ = top k eigenvectors of A, up to rotation

Intuitions & toy example

- Recall: A is adjacency matrix of positive pair probabilities
- $\hat{F} = \mathrm{argmin}_{\hat{F}} || A - \hat{F}\hat{F}^T ||_F$ if we use spectral contrastive loss (HaoChen et al. 2021)
- Columns of $\hat{F}$ = top k eigenvectors of A, up to rotation
- The eigenvalues of $\hat{F}$ (and their ranking) are a function of connectivity parameters $\alpha, \beta, \gamma, \rho$

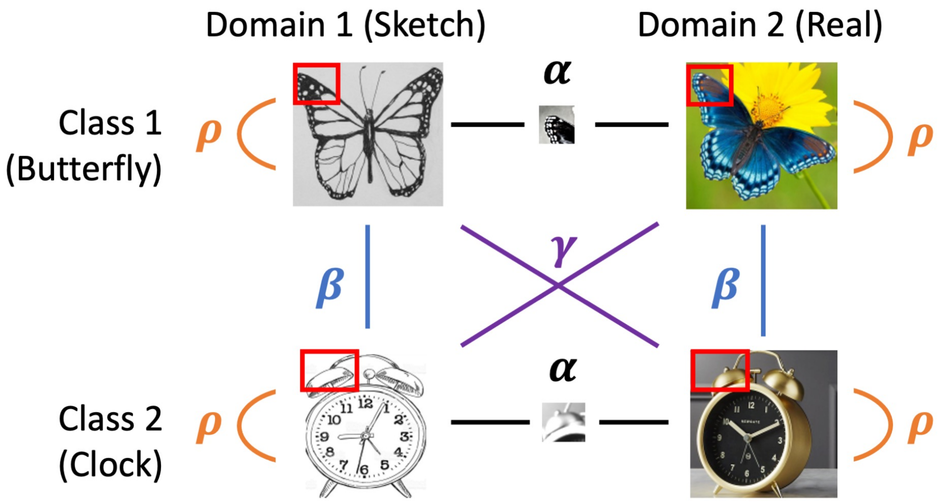
Intuitions & toy example

- Recall: A is adjacency matrix of positive pair probabilities
- $\hat{F} = \text{argmin}_{\hat{F}} || A - \hat{F}\hat{F}^T ||_F$ if we use spectral contrastive loss (HaoChen et al. 2021)
- Columns of $\hat{F}$ = top k eigenvectors of A, up to rotation
- The eigenvalues of $\hat{F}$ (and their ranking) are a function of connectivity parameters $\alpha, \beta, \gamma, \rho$

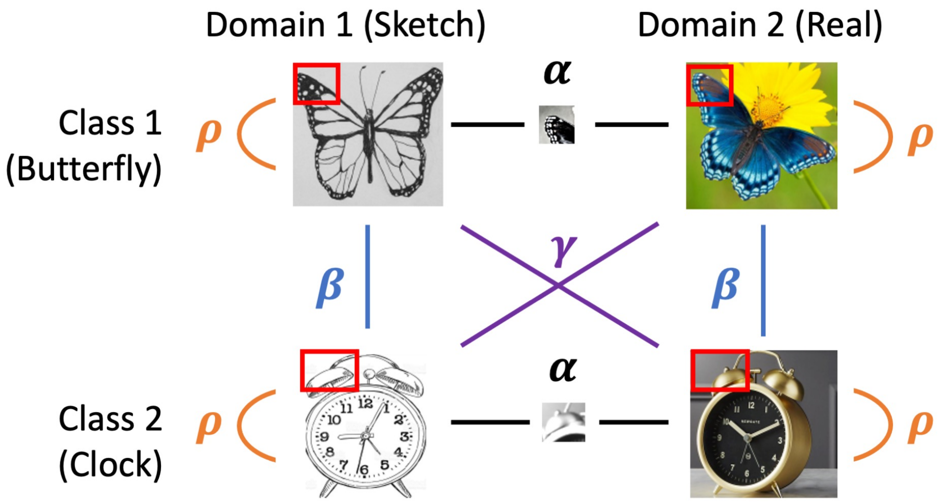
$\Longrightarrow$ connectivity parameters control pre-trained features $\hat{F}$

Intuitions & toy example

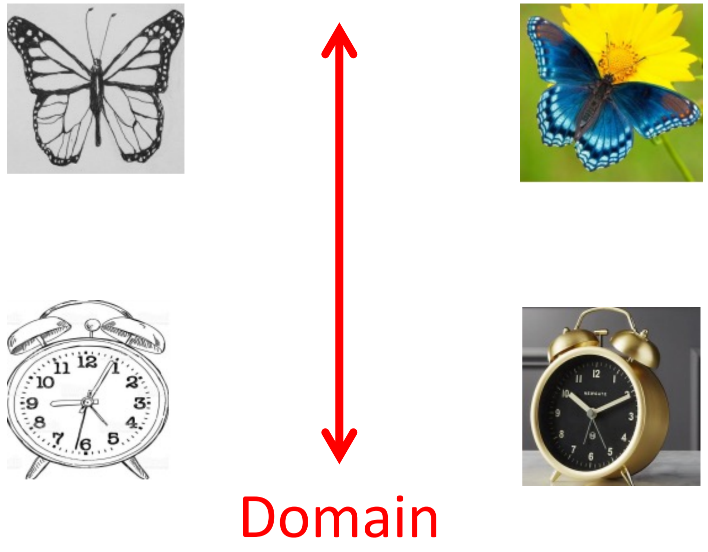
If $\mathbf{min}(\boldsymbol{\alpha}, \boldsymbol{\beta}) > \boldsymbol{\gamma}$ (and self-loop ρ is the largest):



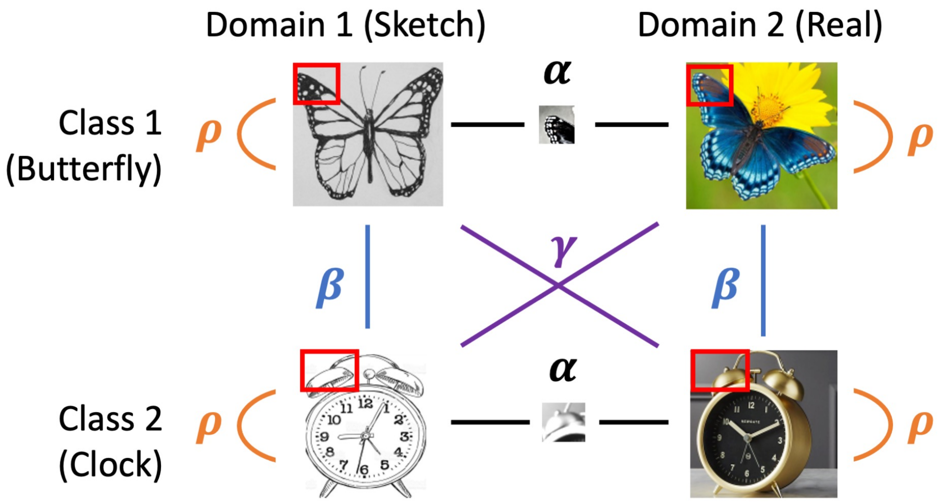
Intuitions & toy example



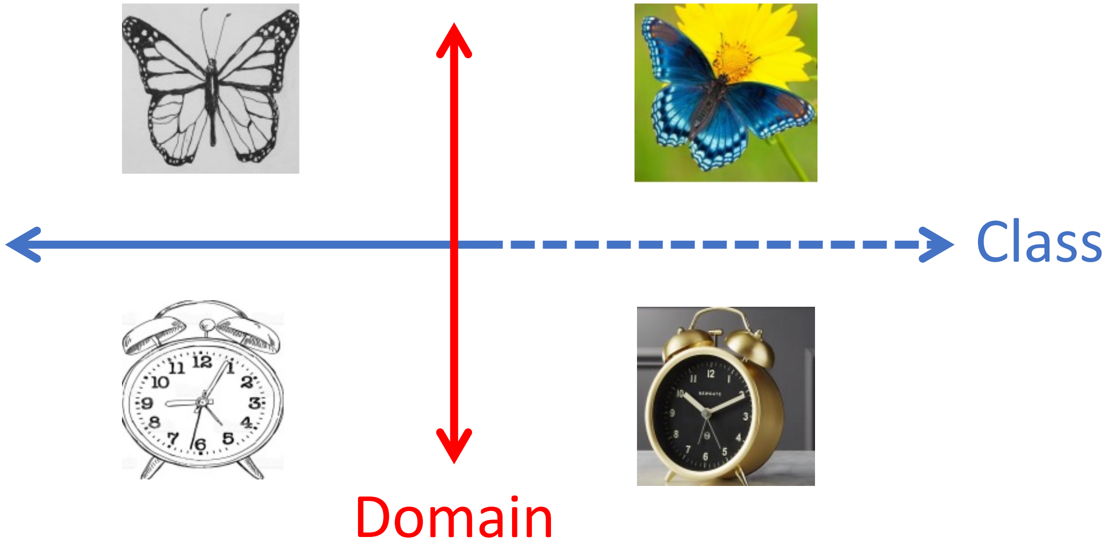
If $\mathbf{min}(\boldsymbol{\alpha}, \boldsymbol{\beta}) > \boldsymbol{\gamma}$ (and self-loop ρ is the largest):



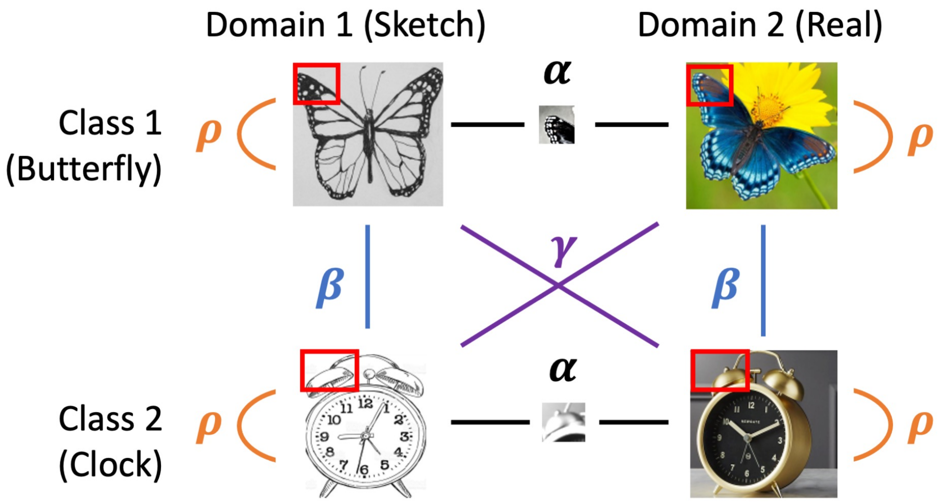
Intuitions & toy example



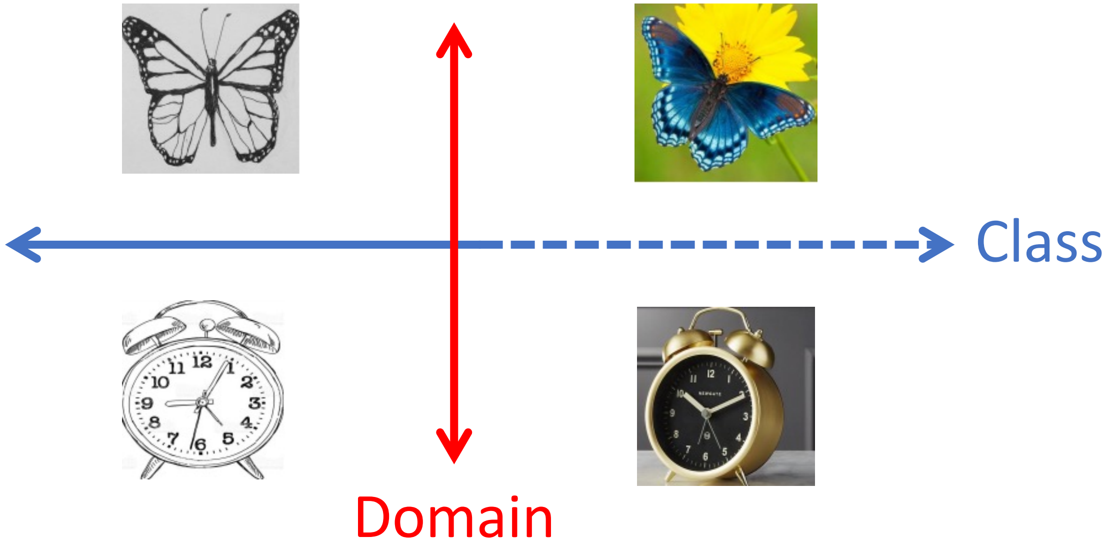
If $\mathbf{min}(\boldsymbol{\alpha}, \boldsymbol{\beta}) > \boldsymbol{\gamma}$ (and self-loop ρ is the largest):



Intuitions & toy example

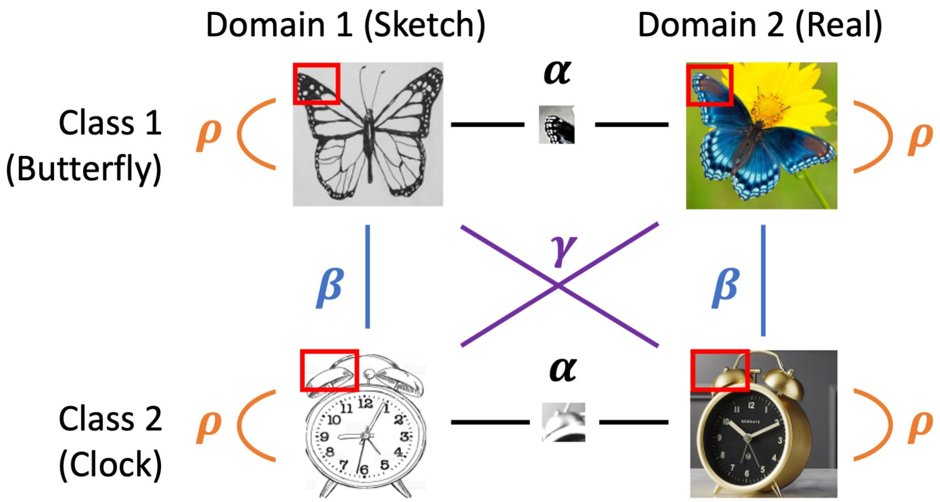


If $\mathbf{min}(\boldsymbol{\alpha}, \boldsymbol{\beta}) > \boldsymbol{\gamma}$ (and self-loop ρ is the largest):

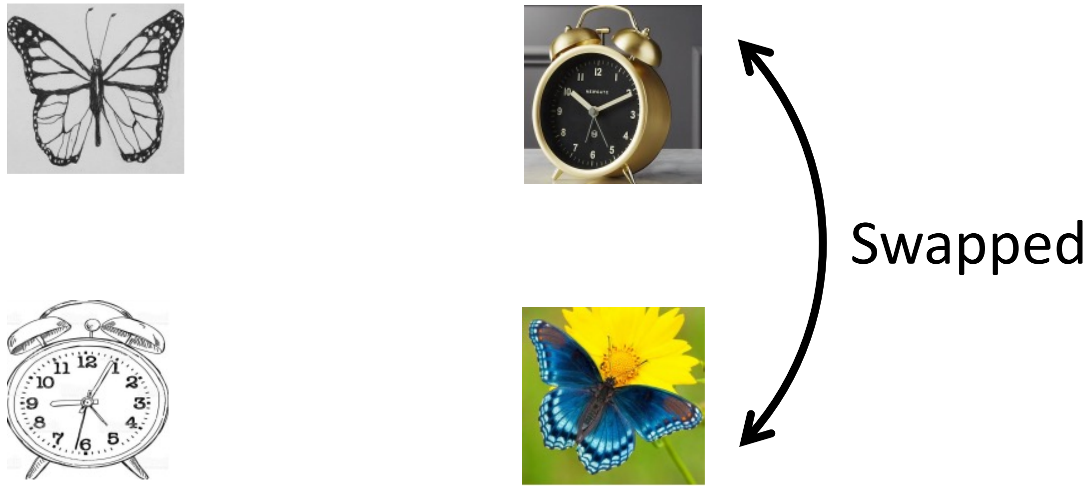


Key condition for transfer: augmentations are more likely to change **only domain** (α) or **only class** (β) than **both domain and class** (γ)

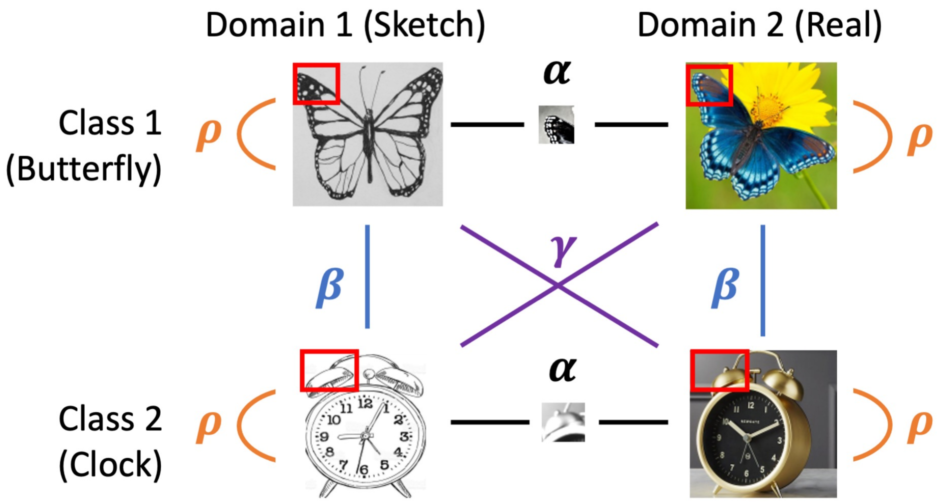
Intuitions & toy example



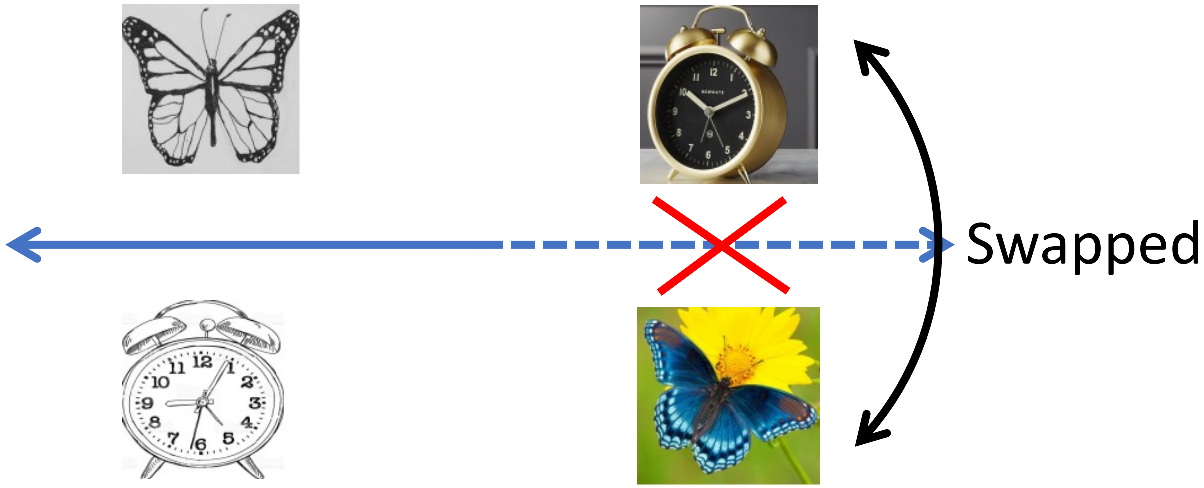
If instead $\alpha < \gamma$:



Intuitions & toy example



If instead $\alpha < \gamma$:



If the condition is violated, the target features can be "swapped" so that a source-trained linear classifier fails to generalize

Generalization beyond simple example

- Consider stochastic block model (SBM): extends to multiple domains, multiple classes, and multiple examples per class/domain

Generalization beyond simple example

- Consider stochastic block model (SBM): extends to multiple domains, multiple classes, and multiple examples per class/domain
- We prove: **same conditions** ($\min(\alpha, \beta) > \gamma$ and ρ is largest) allow contrastive pre-training to learn linearly transferable features (with easily separable source and target features)

Generalization beyond simple example

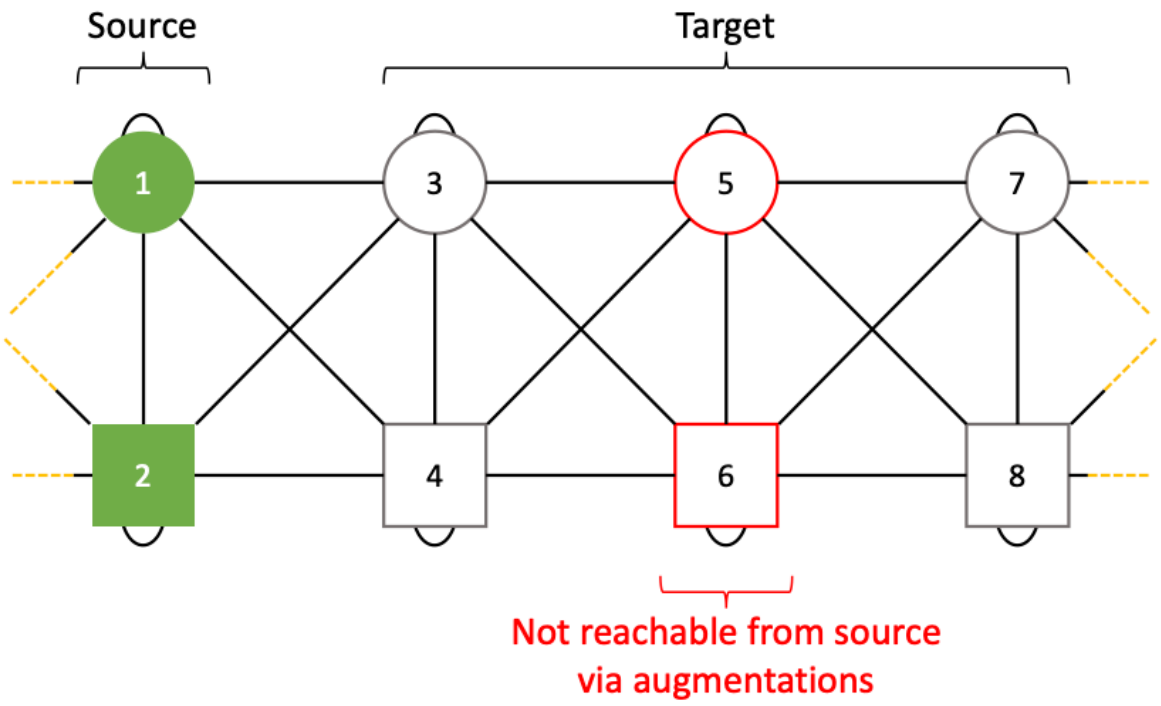
- Consider stochastic block model (SBM): extends to multiple domains, multiple classes, and multiple examples per class/domain
- We prove: **same conditions** ($\min(\alpha, \beta) > \gamma$ and ρ is largest) allow contrastive pre-training to learn linearly transferable features (with easily separable source and target features)
- Follow-up work generalizes beyond random graph models: HaoChen et al. 2022

Contrastive pre-training vs. baselines

- We give an instance where contrastive pre-training can outperform ERM and DANN, even with the same augmentations

Contrastive pre-training vs. baselines

- We give an instance where contrastive pre-training can outperform ERM and DANN, even with the same augmentations



Outline

- Setup: augmentation graph
- Intuitions and theoretical results
  - Main intuitions (toy example)
  - Results for stochastic block model & beyond
  - Contrastive pre-training vs. ERM & DANN
- Test theoretical predictions on real data

Connectivity predicts target accuracy

- Our theory predicts that target accuracy depends on α, β, γ and requires that $\alpha > \gamma$ and $\beta > \gamma$

Connectivity predicts target accuracy

- Our theory predicts that target accuracy depends on α, β, γ and requires that $\alpha > \gamma$ and $\beta > \gamma$
- Estimate α, β, γ by training a classifier to predict between augmented images of different domains/classes, evaluate on held out examples

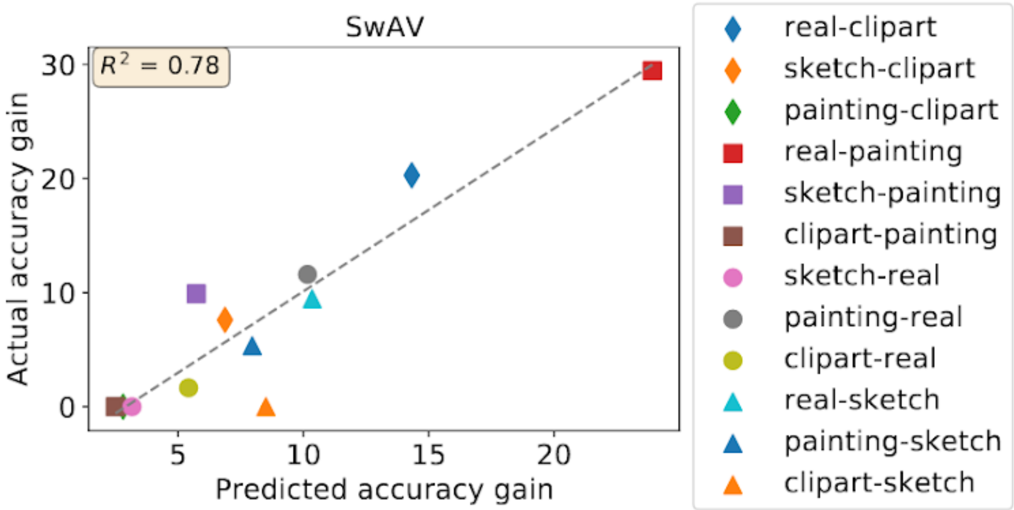
Connectivity predicts target accuracy

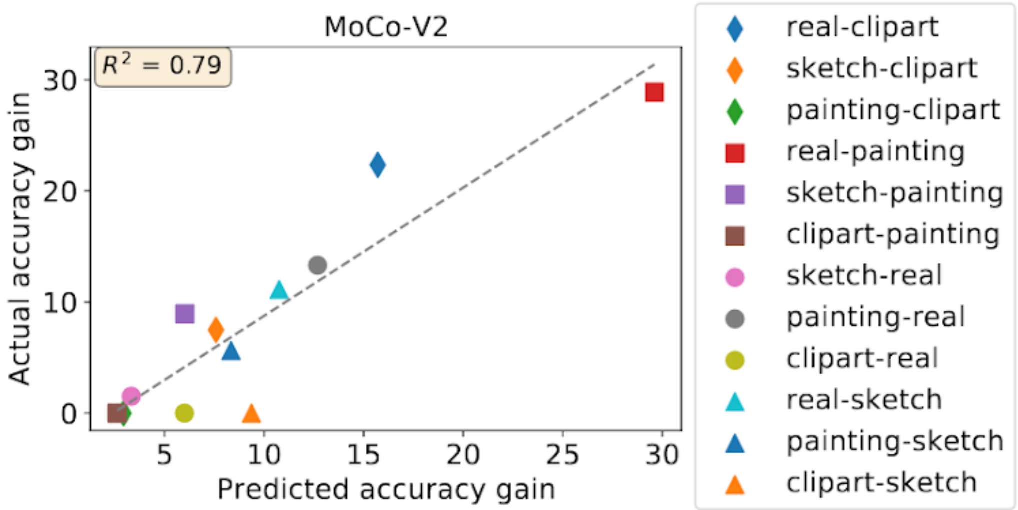
- Our theory predicts that target accuracy depends on α, β, γ and requires that $\alpha > \gamma$ and $\beta > \gamma$
- Estimate α, β, γ by training a classifier to predict between augmented images of different domains/classes, evaluate on held out examples

$$\text{target accuracy} \approx (\alpha/\gamma)^{w_1} \cdot (\beta/\gamma)^{w_2}$$

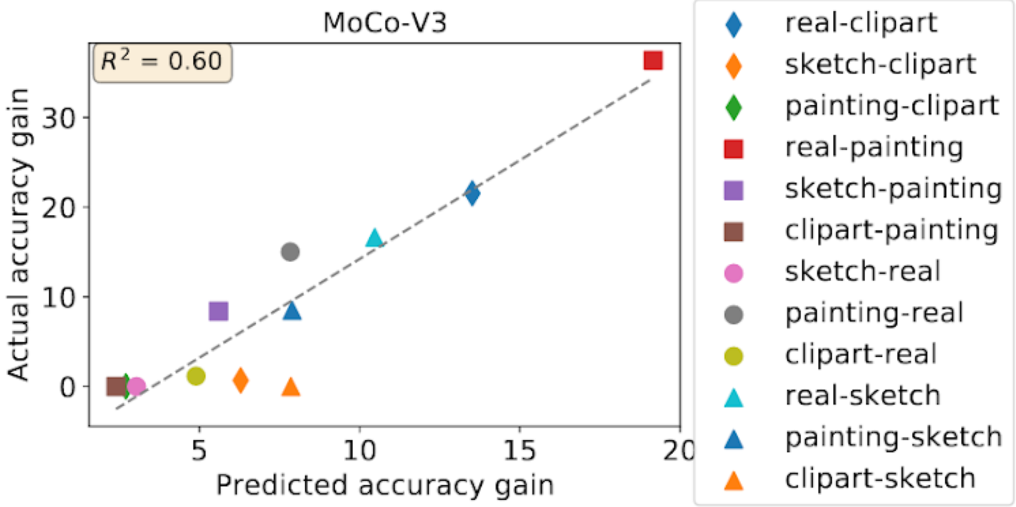
- Estimate w_1, w_2 by fitting a linear function in log space and determine quality of fit compared to a control

Predicting target accuracy (contrastive methods)

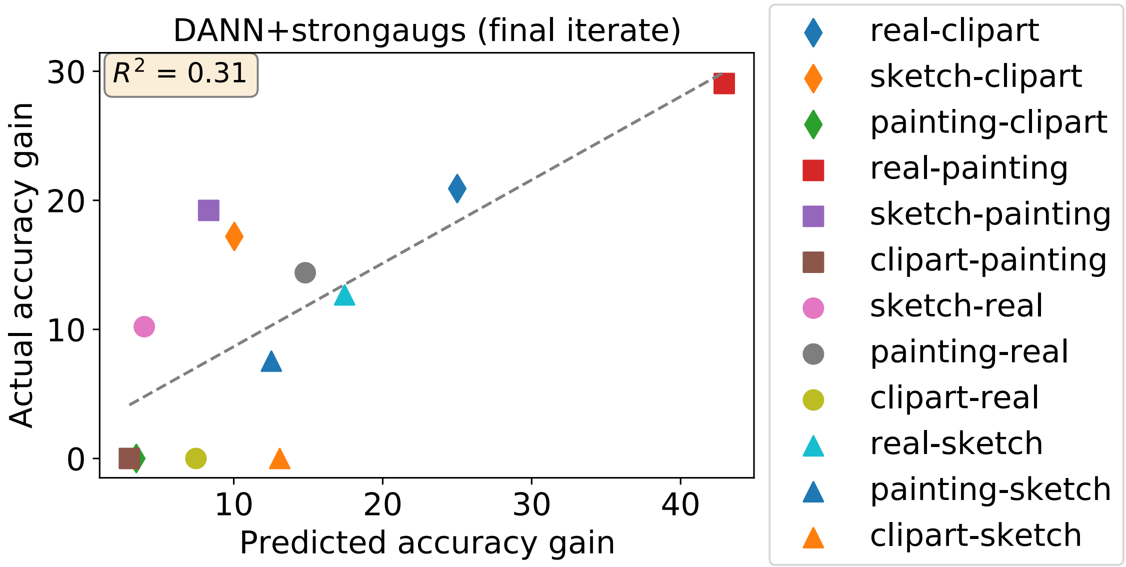


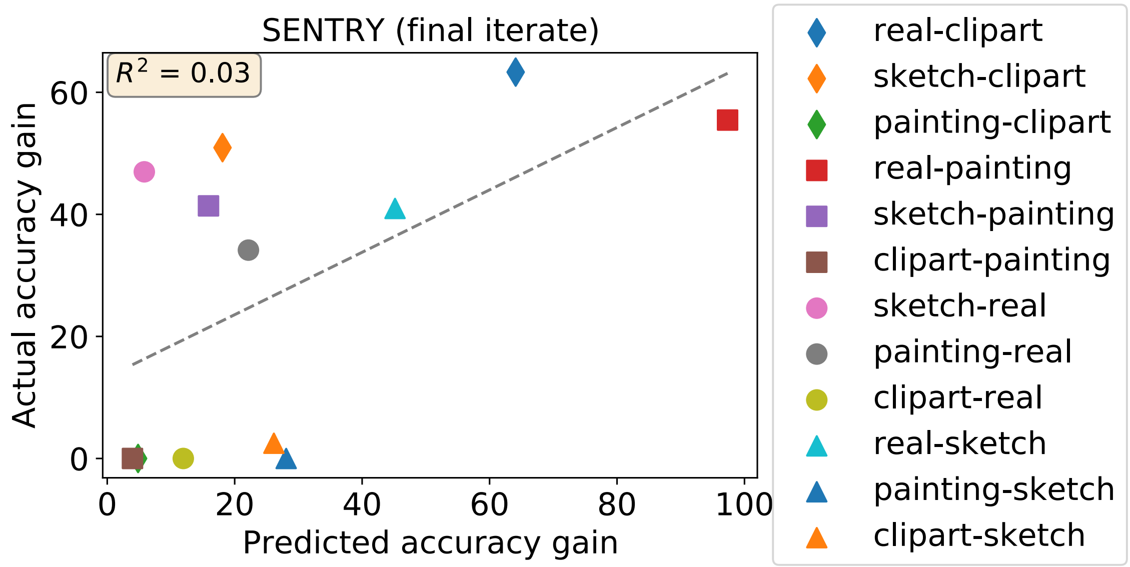


Method	R^2
SwAV	0.78
MoCo-V2	0.79
MoCo-V3	0.60



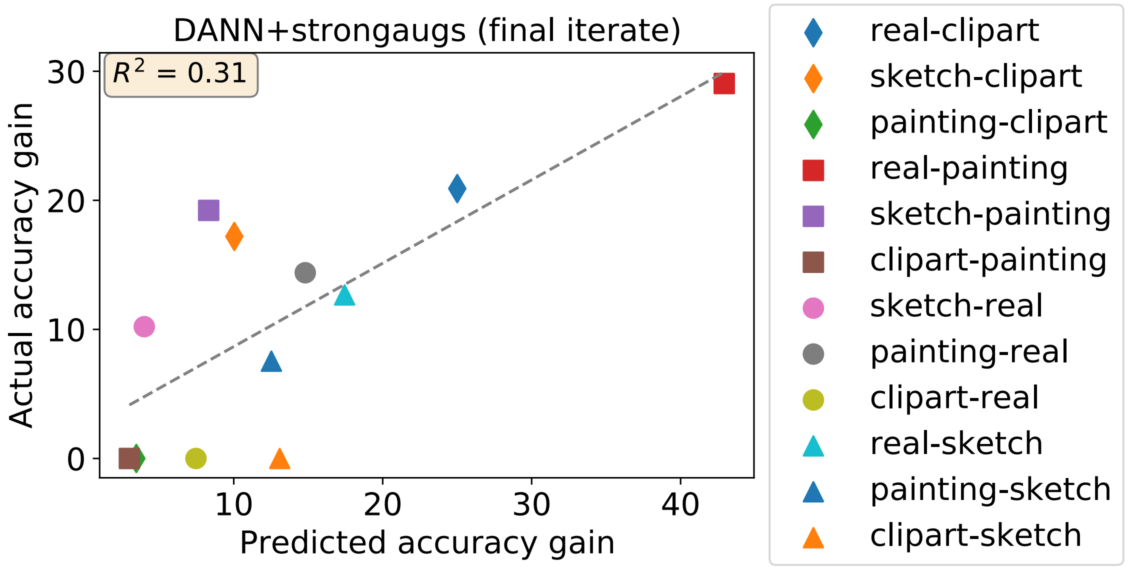
Predicting target accuracy (controls)

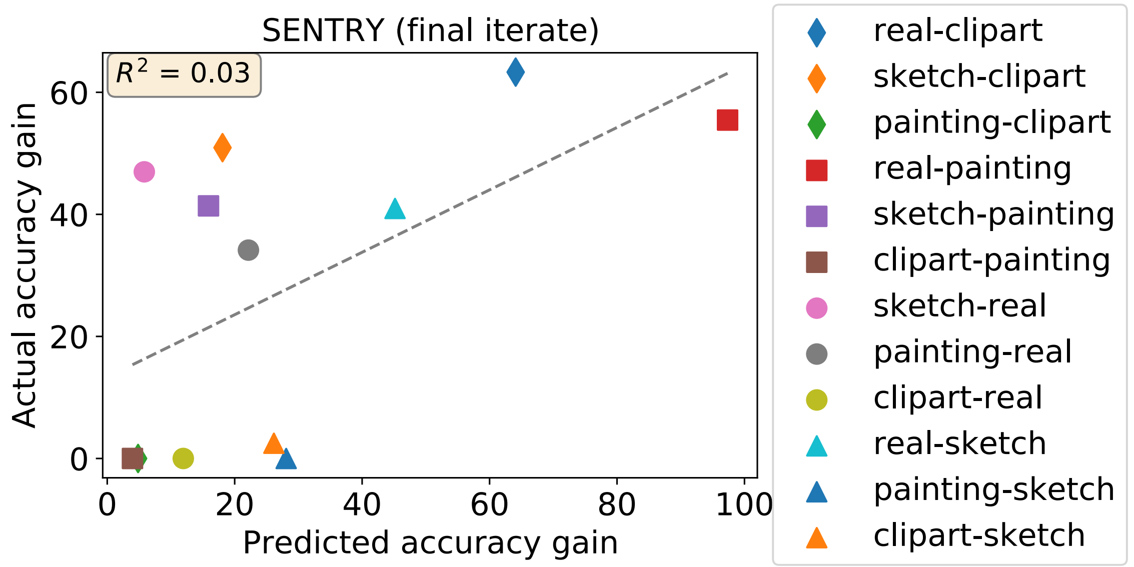




Method	R^2
SwAV	0.78
MoCo-V2	0.79
MoCo-V3	0.60

Predicting target accuracy (controls)





Method	R^2
SwAV	0.78
MoCo-V2	0.79
MoCo-V3	0.60
DANN	0.31
SENTRY	0.03

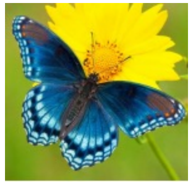
Lower quality of fit for non-contrastive methods: DANN and SENTRY

Class and domain are disentangled

- We train a linear probe for class and domain information in the contrastive features, finding that class and domain classifiers have low cosine similarity

Class and domain are disentangled

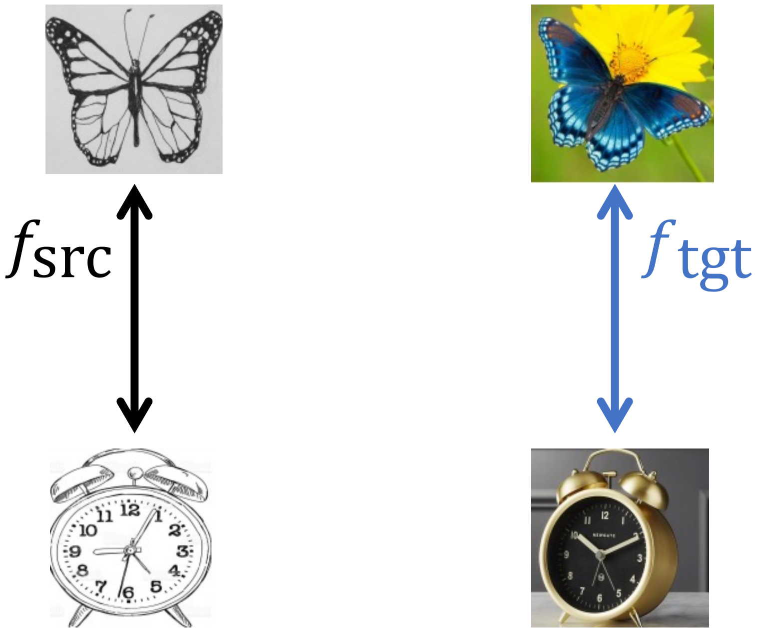
- We train a linear probe for class and domain information in the contrastive features, finding that class and domain classifiers have low cosine similarity

	f_{src} vs. f_{tgt}	f_{src} vs. f_{dom}	f_{tgt} vs. f_{dom}
Living-17	0.397	0.013	0.016
DomainNet	0.187	0.018	0.018


Class and domain are disentangled

- We train a linear probe for class and domain information in the contrastive features, finding that class and domain classifiers have low cosine similarity

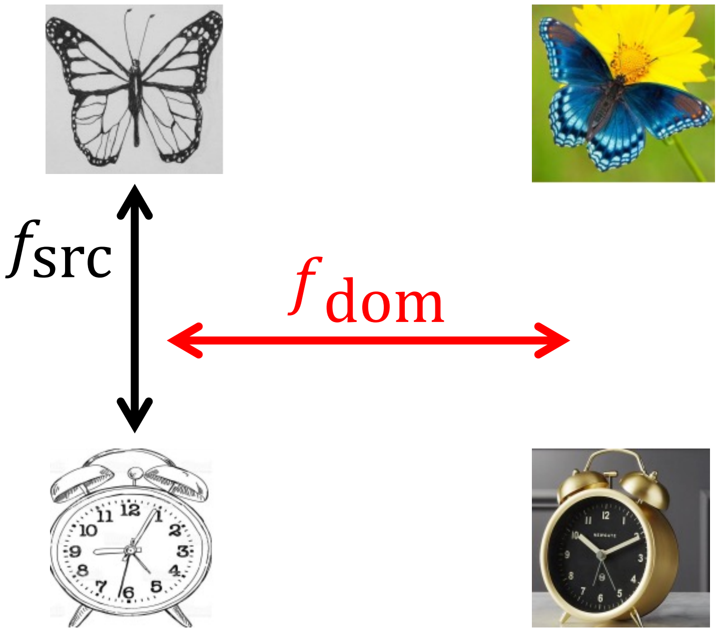


	f_{src} vs. f_{tgt}	f_{src} vs. f_{dom}	f_{tgt} vs. f_{dom}
Living-17	0.397	0.013	0.016
DomainNet	0.187	0.018	0.018

Aligned

Class and domain are disentangled

- We train a linear probe for class and domain information in the contrastive features, finding that class and domain classifiers have low cosine similarity

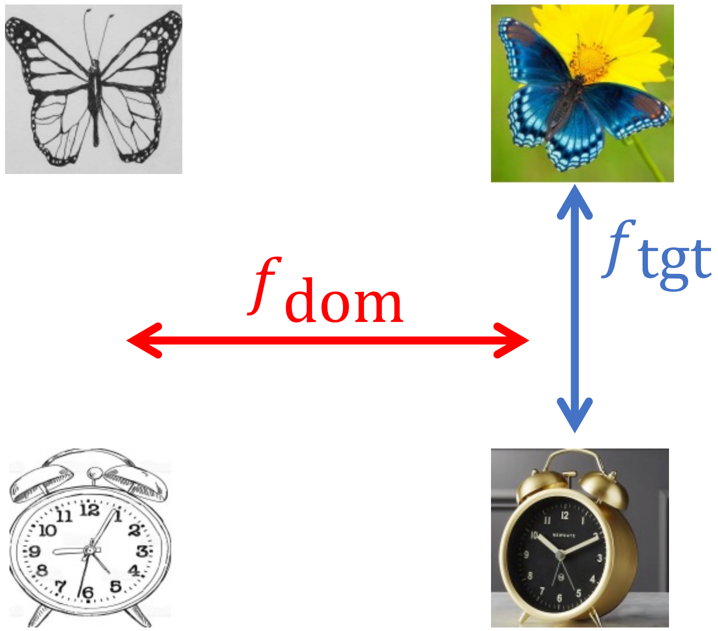


	f_{src} vs. f_{tgt}	f_{src} vs. f_{dom}	f_{tgt} vs. f_{dom}
Living-17	0.397	0.013	0.016
DomainNet	0.187	0.018	0.018

Orthogonal

Class and domain are disentangled

- We train a linear probe for class and domain information in the contrastive features, finding that class and domain classifiers have low cosine similarity



	f_{src} vs. f_{tgt}	f_{src} vs. f_{dom}	f_{tgt} vs. f_{dom}
Living-17	0.397	0.013	0.016
DomainNet	0.187	0.018	0.018

Orthogonal

Dropping examples

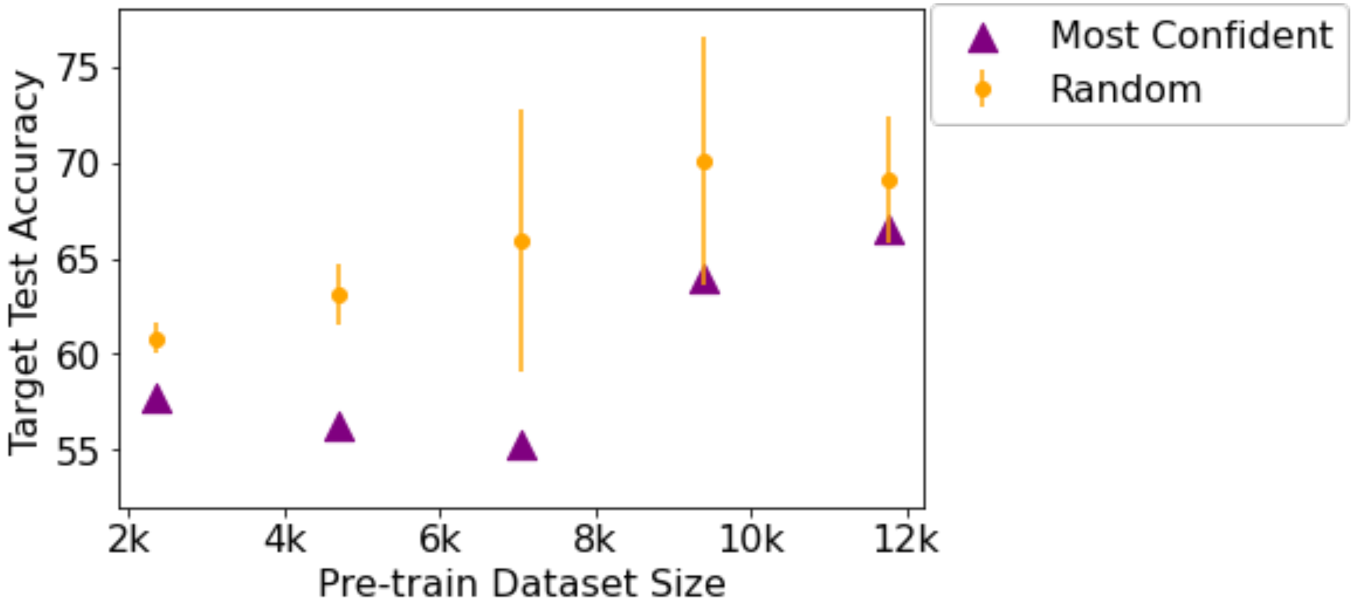
- Consider random examples vs. examples that are “in between” two domains, contributing most to connectivity between domains

Dropping examples

- Consider random examples vs. examples that are “in between” two domains, contributing most to connectivity between domains
- Procedure: train classifier to distinguish between source and target domains, pre-train only on examples classifier is most confident on

Dropping examples

- Consider random examples vs. examples that are “in between” two domains, contributing most to connectivity between domains
- Procedure: train classifier to distinguish between source and target domains, pre-train only on examples classifier is most confident on



Target Unlabeled Data is Important

- Access to target unlabeled examples is important for robustness (pretraining on source examples alone does not lead to robustness gains)

Target Unlabeled Data is Important

- Access to target unlabeled examples is important for robustness (pretraining on source examples alone does not lead to robustness gains)

	ERM	SwAV (S)	SwAV (T)	**SwAV (S+T)**
Living-17	63.29	62.71	70.41	**75.12**
Entity-30	52.52	52.33	60.33	**62.03**

Conclusion

- Contrastive pre-training is a competitive method for UDA

Conclusion

- Contrastive pre-training is a competitive method for UDA
- Works without collapsing source and target representations

Conclusion

- Contrastive pre-training is a competitive method for UDA
- Works without collapsing source and target representations
- Instead, disentangles class and domain information, enabling transfer
  - Consequence of the structure of connections between domains and classes via data augmentations