Domain Generalization via Nuclear Norm Regularization

Zhenmei Shi*, Yifei Ming*, Ying Fan*, Frederic Sala, Yingyu Liang
University of Wisconsin-Madison

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

SAPIENTIA·ET·VIRTUS

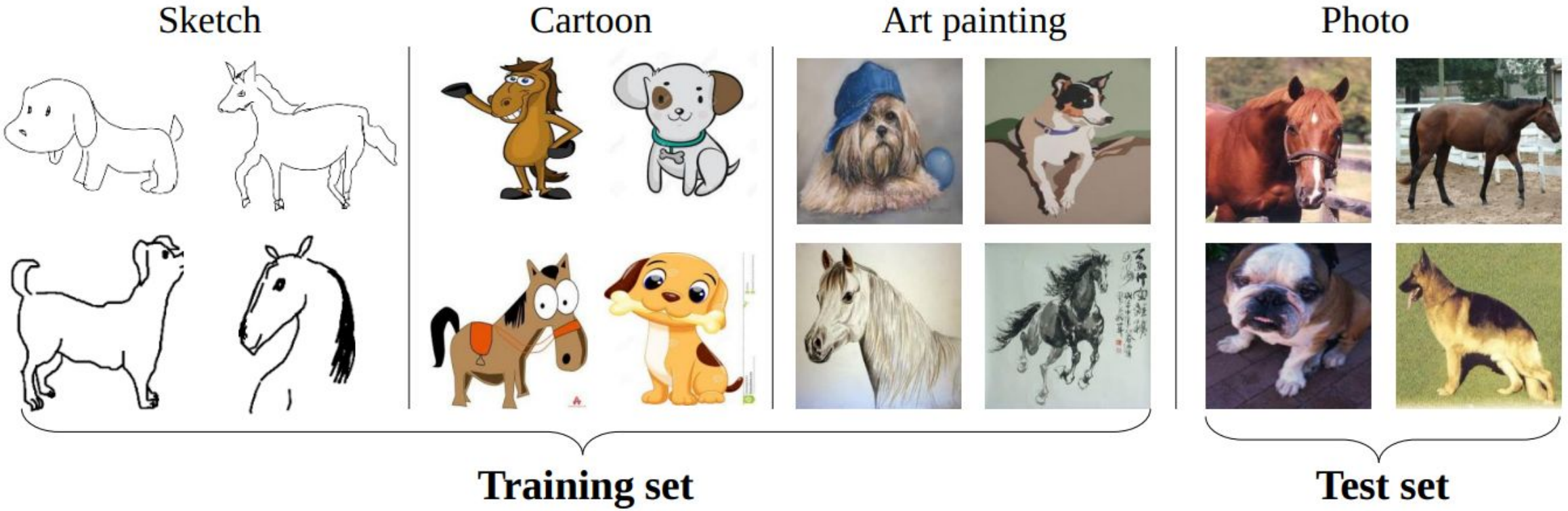
CPAL 2024

CPAL
LESS IS MORE

Intro - Domain Generalization

Train on multiple training domains, e.g., Sketch + Cartoon + Art.

Test on **new/unseen** domain, e.g., Photo.

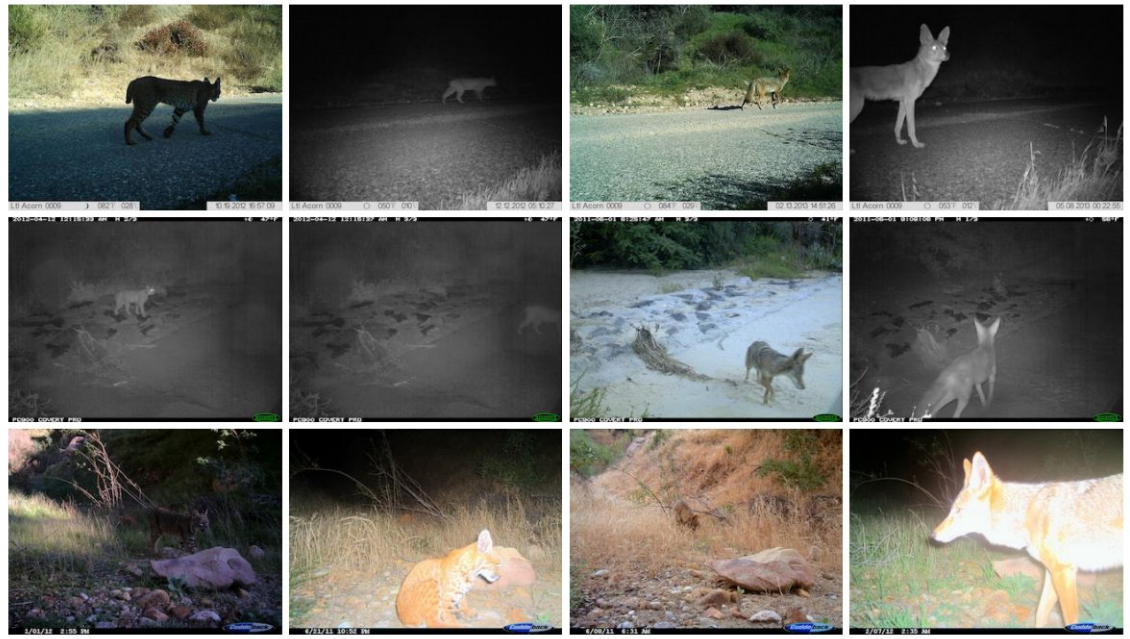


PACS dataset

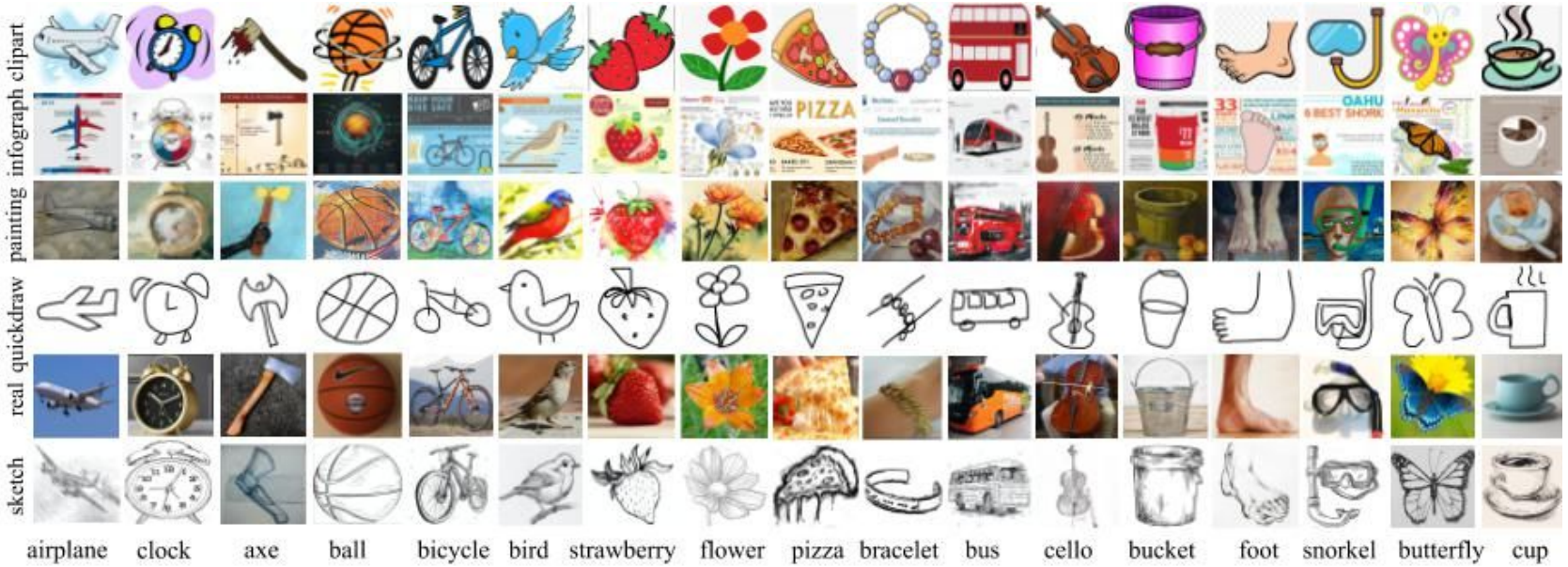
Source: Deeper, broader and artier domain generalization. ICCV 2017.

Intro - More Datasets

OfficeHome



Terra Incognita

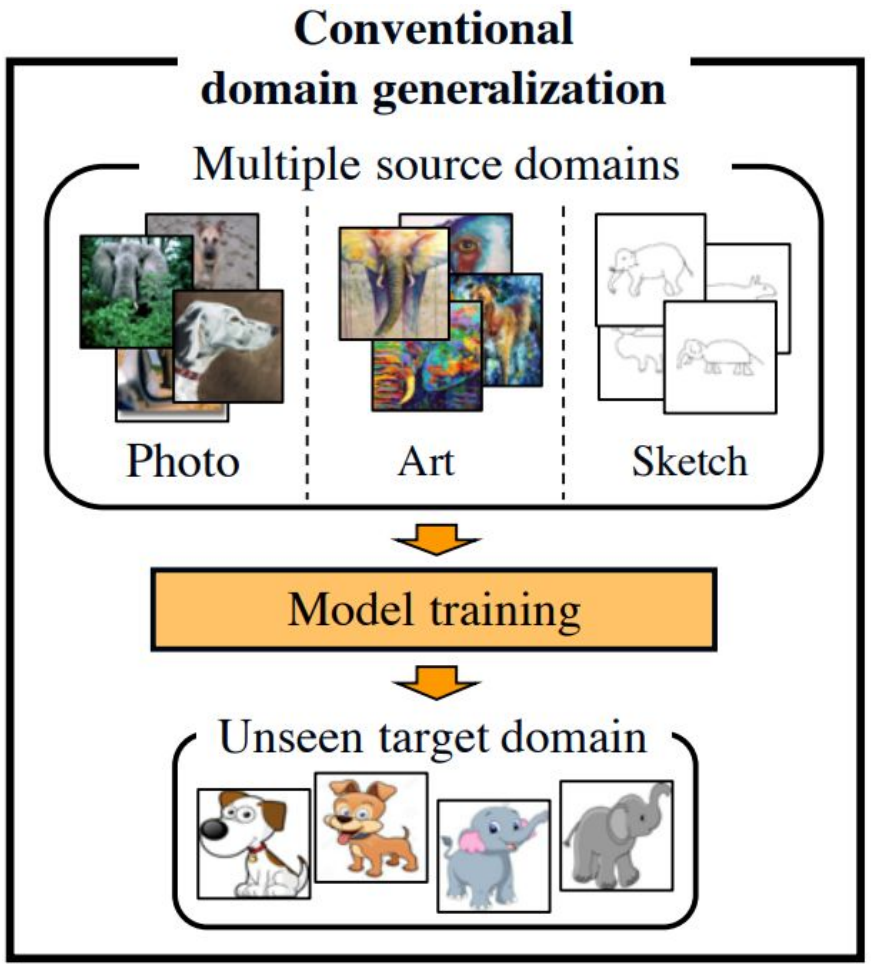


DomainNet

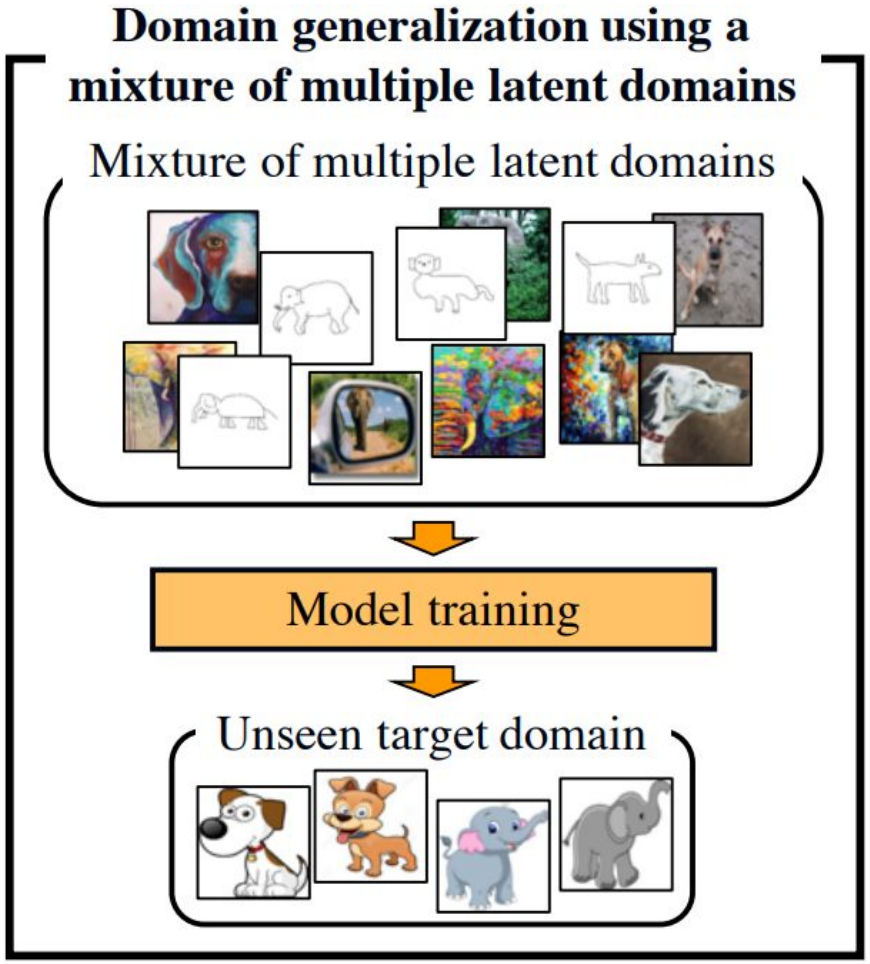
6 domains,
345 classes,
586,575 images

Source: In Search of Lost Domain Generalization. ICLR 2021.

Intro - Domain Labels



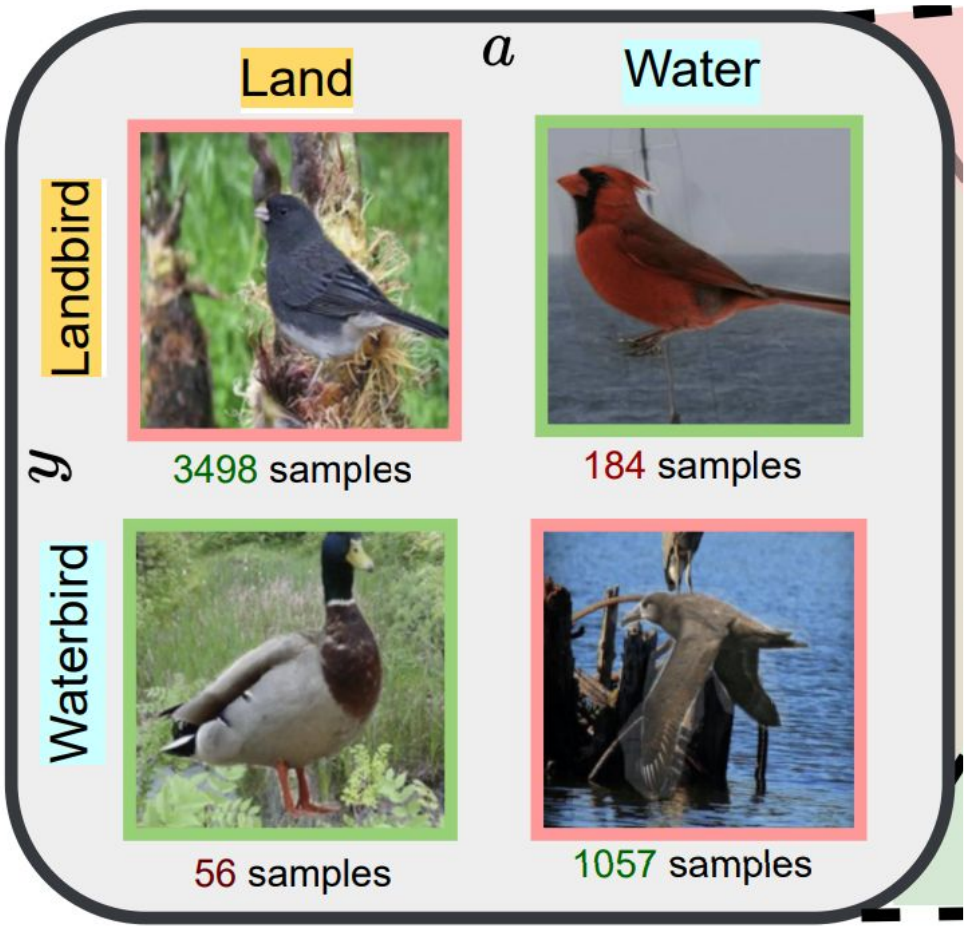
With domain labels



Without domain labels (ours)

Source: Domain Generalization Using a Mixture of Multiple Latent Domains. AAAI 2020.

Intro - Invariant/Spurious Feature

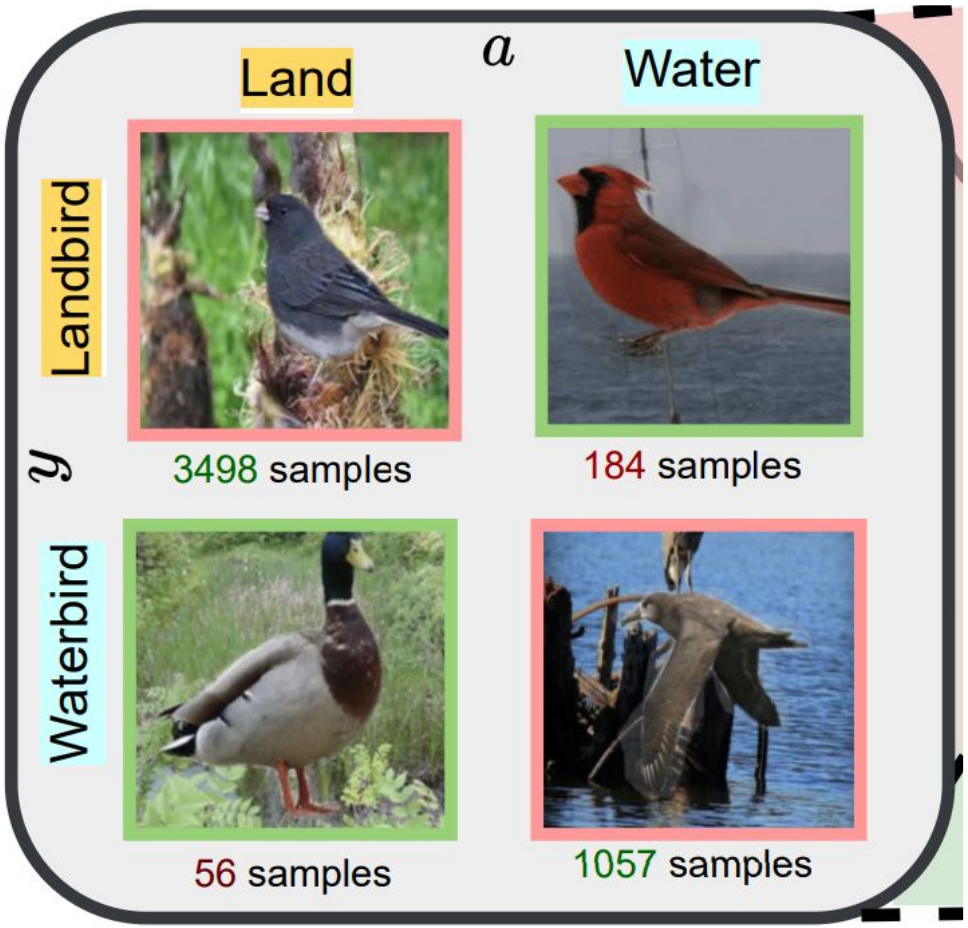


Waterbirds dataset

Invariant - Birds; Spurious - Background

Source: Avoiding spurious correlations via logit correction, ICLR 2023

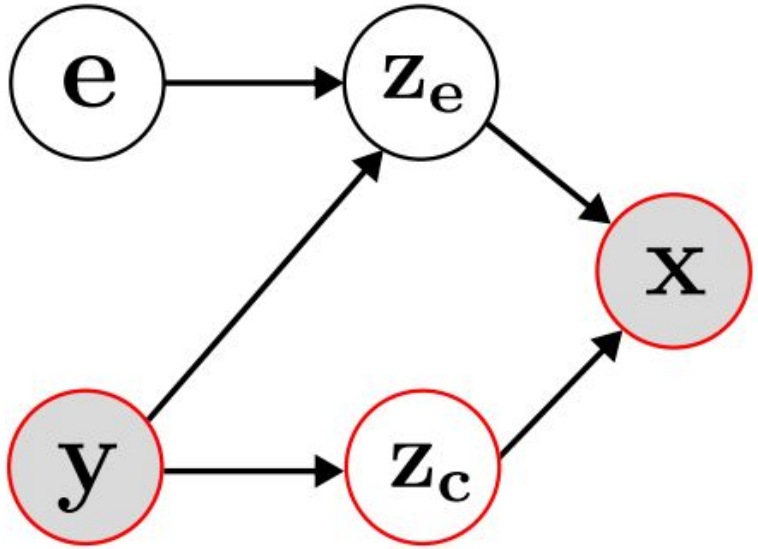
Intro - Invariant/Spurious Feature



Waterbirds dataset

Invariant - Birds; Spurious - Background

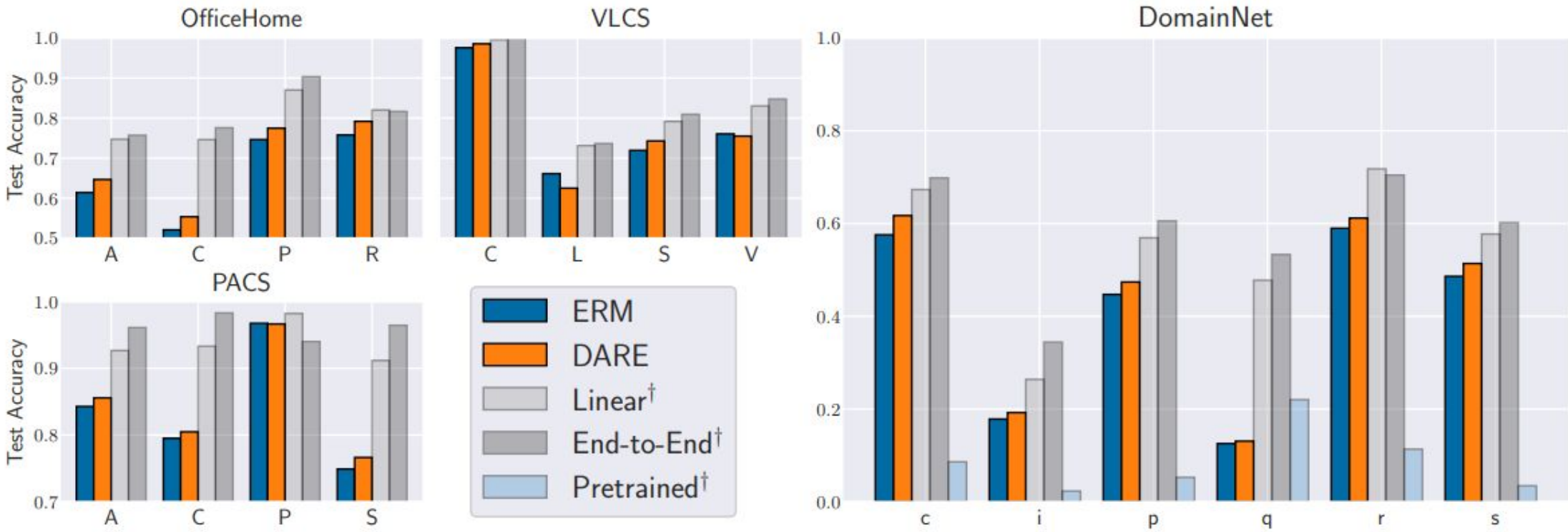
Source: Avoiding spurious correlations via logit correction, ICLR 2023



Hidden representation data model:

- e : environment (background)
- y : label (bird)
- z_e: spurious feature
- z_c: invariant feature
- x : input (image)

Motivation - ERM Learn Good Features



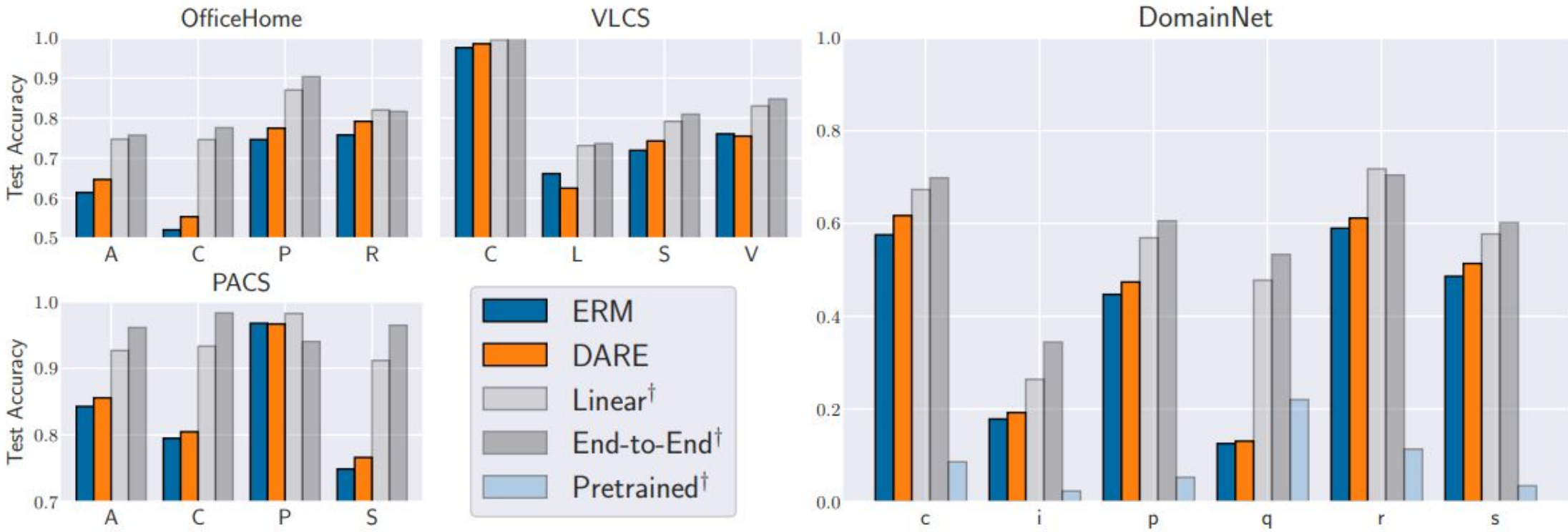
Empirical Risk Minimization already learn features sufficient for domain generalization:

- ERM : train on training domains.
- Linear : train on training domains => Linear Probing on **unseen** domain.
- End-to-End: train on training + **unseen** domain.

Evaluate on the unseen domain.

Source: Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization, 2023

Motivation - ERM Learn Good Features



- **Main Issue:** features in ERM can be arbitrarily mixed: spurious features are hard to disentangle from invariant features.
- **Idea:** low-dimensional (parsimonious) structures => minimal information retrieved from ERM solution from training domains by controlling the **rank** => avoid domain overfitting.
- **Hypothesis**: spurious features have lower correlation with labels than invariant features.

Source: Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization, 2023

Question

Question:

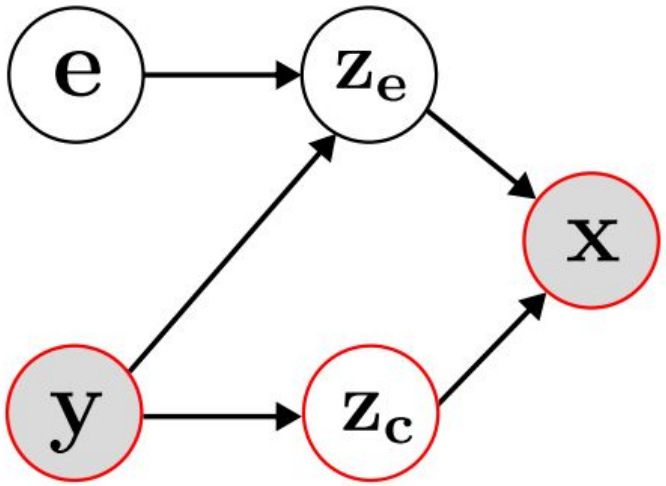
Can ERM benefit from rank regularization of the extracted feature for better domain generalization?

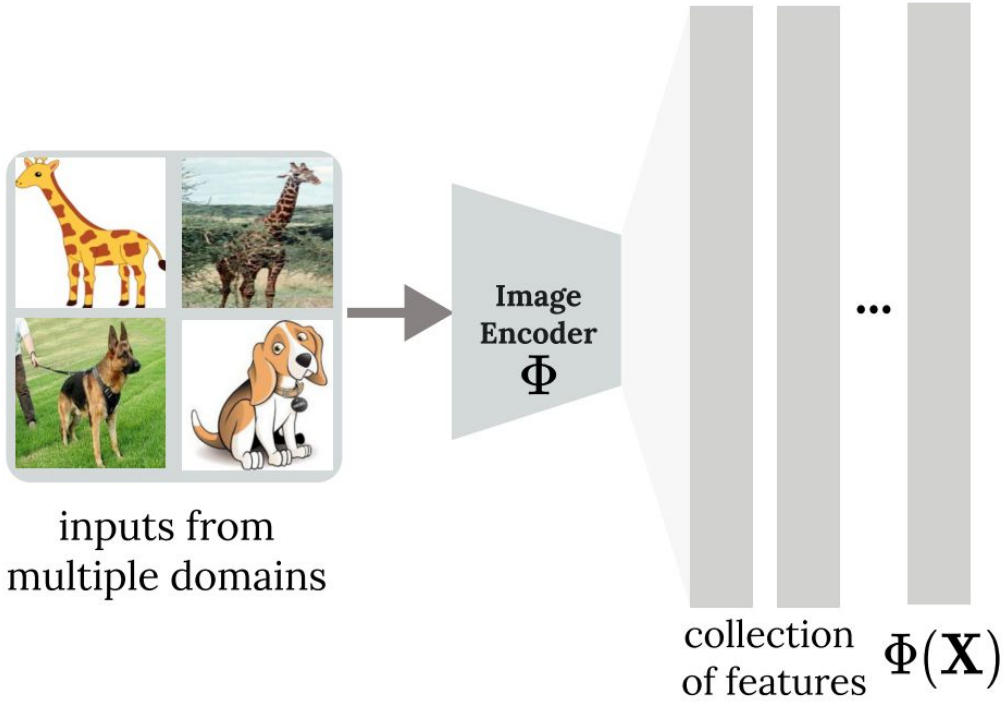
Answer:

Yes, ERM with Nuclear Norm Regularization (ERM-NU).

Nuclear norm is convex envelope to the rank function.

Method - Setup





Hidden representation data model:

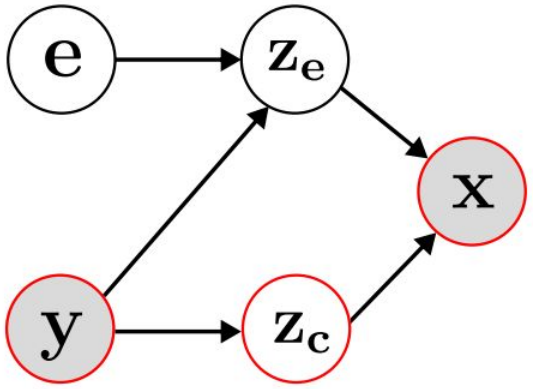
- e : environment (background)
- y : label (bird)
- z_e: spurious feature
- z_c: invariant feature
- x : input (image)

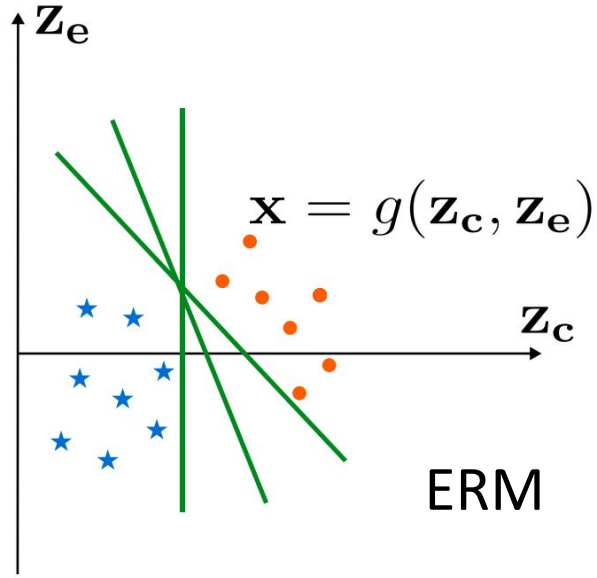
a : linear head

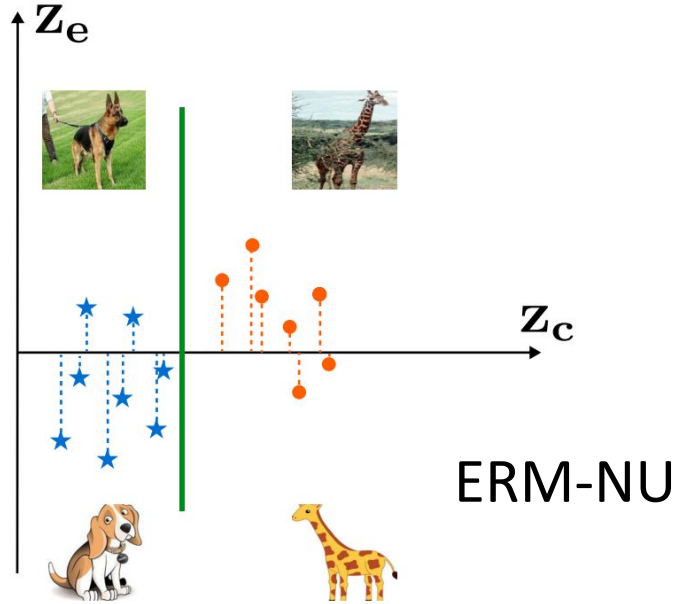
Φ: feature extractor, e.g., ResNet 50

Method - Objective Function

$$\min_{\mathbf{a},\Phi} \underbrace{\mathcal{L}(\mathbf{a}, \Phi)}_{\text{ERM}} + \lambda \underbrace{\|\Phi(\mathbf{X})\|_*}_{\text{NU}}$$







NU can select a subset of ERM solutions that extract the smallest possible information for classification => reduce the effect of spurious features for better generalization.

Experiment - Simulation

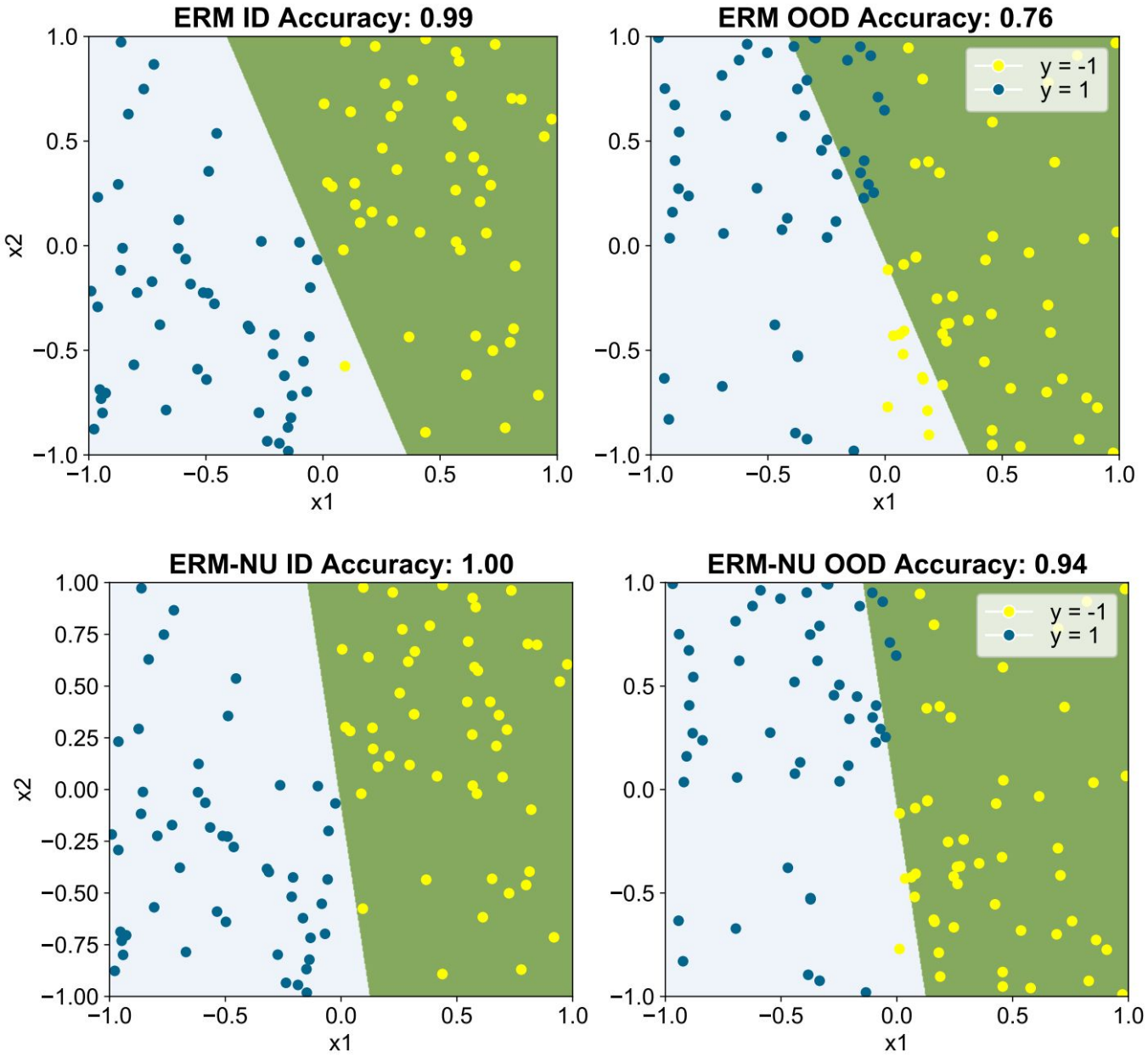
In-distribution (ID) / training:

- x_1 and y has strong correlation.
- x_2 and y has weak correlation.

Out-of-distribution (OOD) / unseen:

- x_1 and y has the same correlation.
- x_2 and y change the correlation.

NU significantly reduces the OOD error rate, while keep small ID error.



Experiment - Real Dataset

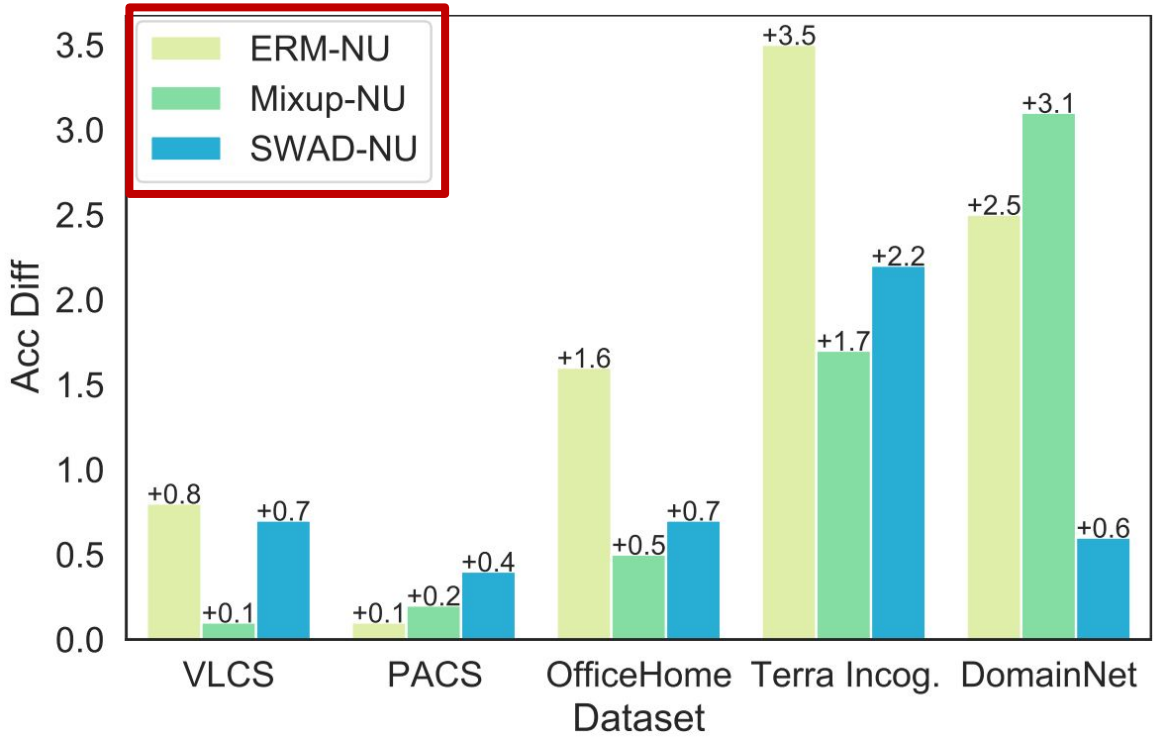
Algorithm	VLCS	PACS	OfficeHome	TerraInc	DomainNet	Average
MMD† (CVPR 18) [10]	77.5 ± 0.9	84.6 ± 0.5	66.3 ± 0.1	42.2 ± 1.6	23.4 ± 9.5	58.8
Mixstyle‡ (ICLR 21) [27]	77.9 ± 0.5	85.2 ± 0.3	60.4 ± 0.3	44.0 ± 0.7	34.0 ± 0.1	60.3
GroupDRO† (ICLR 19) [28]	76.7 ± 0.6	84.4 ± 0.8	66.0 ± 0.7	43.2 ± 1.1	33.3 ± 0.2	60.7
IRM† (ArXiv 20) [6]	78.5 ± 0.5	83.5 ± 0.8	64.3 ± 2.2	47.6 ± 0.8	33.9 ± 2.8	61.6
ARM† (ArXiv 20) [29]	77.6 ± 0.3	85.1 ± 0.4	64.8 ± 0.3	45.5 ± 0.3	35.5 ± 0.2	61.7
VREx† (ICML 21) [14]	78.3 ± 0.2	84.9 ± 0.6	66.4 ± 0.6	46.4 ± 0.6	33.6 ± 2.9	61.9
CDANN† (ECCV 18) [8]	77.5 ± 0.1	82.6 ± 0.9	65.8 ± 1.3	45.8 ± 1.6	38.3 ± 0.3	62.0
AND-mask* (ICLR 20)[30]	78.1 ± 0.9	84.4 ± 0.9	65.6 ± 0.4	44.6 ± 0.3	37.2 ± 0.6	62.0
DANN† (JMLR 16) [7]	78.6 ± 0.4	83.6 ± 0.4	65.9 ± 0.6	46.7 ± 0.5	38.3 ± 0.1	62.6
RSC† (ECCV 20) [31]	77.1 ± 0.5	85.2 ± 0.9	65.5 ± 0.9	46.6 ± 1.0	38.9 ± 0.5	62.7
MTL† (JMLR 21) [32]	77.2 ± 0.4	84.6 ± 0.5	66.4 ± 0.5	45.6 ± 1.2	40.6 ± 0.1	62.9
Mixup† (ICLR 18) [1]	77.4 ± 0.6	84.6 ± 0.6	68.1 ± 0.3	47.9 ± 0.8	39.2 ± 0.1	63.4
MLDG† (AAAI 18) [33]	77.2 ± 0.4	84.9 ± 1.0	66.8 ± 0.6	47.7 ± 0.9	41.2 ± 0.1	63.6
Fish (ICLR 22) [34]	77.8 ± 0.3	85.5 ± 0.3	68.6 ± 0.4	45.1 ± 1.3	42.7 ± 0.2	63.9
Fishr* (ICML 22) [35]	77.8 ± 0.1	85.5 ± 0.4	67.8 ± 0.1	47.4 ± 1.6	41.7 ± 0.0	64.0
SagNet† (CVPR 21) [36]	77.8 ± 0.5	86.3 ± 0.2	68.1 ± 0.1	48.6 ± 1.0	40.3 ± 0.1	64.2
SelfReg (ICCV 21) [37]	77.8 ± 0.9	85.6 ± 0.4	67.9 ± 0.7	47.0 ± 0.3	41.5 ± 0.2	64.2
CORAL† (ECCV 16) [9]	78.8 ± 0.6	86.2 ± 0.3	68.7 ± 0.3	47.6 ± 1.0	41.5 ± 0.1	64.5
SAM‡ (ICLR 21) [38]	79.4 ± 0.1	85.8 ± 0.2	69.6 ± 0.1	43.3 ± 0.7	44.3 ± 0.0	64.5
mDSDI (NeurIPS 21) [39]	79.0 ± 0.3	86.2 ± 0.2	69.2 ± 0.4	48.1 ± 1.4	42.8 ± 0.1	65.1
MIRO (ECCV 22) [40]	79.0 ± 0.0	85.4 ± 0.4	70.5 ± 0.4	50.4 ± 1.1	44.3 ± 0.2	65.9
ERM† [41]	77.5 ± 0.4	85.5 ± 0.2	66.5 ± 0.3	46.1 ± 1.8	40.9 ± 0.1	63.3
ERM-NU (ours)	**78.3** ± 0.3	**85.6** ± 0.1	**68.1** ± 0.1	**49.6** ± 0.6	**43.4** ± 0.1	**65.0**
SWAD‡ (NeurIPS 21) [24]	79.1 ± 0.1	88.1 ± 0.1	70.6 ± 0.2	50.0 ± 0.3	46.5 ± 0.1	66.9
SWAD-NU (ours)	**79.8** ± 0.2	**88.5** ± 0.2	**71.3** ± 0.3	**52.2** ± 0.3	**47.1** ± 0.1	**67.8**

SWAD: Domain Generalization by Seeking Flat Minima

NU is effective.

Experiment - Real Dataset

NU is broadly applicable.



NU is easy to implement.

```
def forward(self, x, y):
    f = self.featurizer(x) # get feature embedding
    loss = F.cross_entropy(self.classifier(f), y) # get classification loss
    _,s,_ = torch.svd(f) # singular value decomposition
    loss += self.lambda * torch.sum(s) # add nuclear norm regularization
    return loss
```

Theoretical Analysis

Theorem (Informal; Linear data and linear model)

- The optimal solution for the **ERM-NU** has **high** OOD test accuracy.
- The optimal solution for the **ERM** with/without weight decay has **low** OOD test accuracy (like random guessing).

Proof Intuition:

1. ERM will encode **all** features correlated with labels, even when the correlation is weak (logistic or cross-entropy loss).
2. Larger correlation with label => stronger feature encoding.
3. When OOD has different spurious feature distributions => ERM fails (random guessing).
4. However, ERM-NU will only encode features that have a large correlation with labels (invariant features) => high OOD test accuracy.

Take Home Message

Nuclear Norm Regularization is an

1. effective
2. broadly applicable
3. easy to implement

method for domain generalization.

Q&A

Thanks!