Efficient Domain Generalization via Common-Specific Low-Rank Decomposition*

Vihari Piratla[1,2] Praneeth Netrapalli[2] Sunita Sarawagi[1]

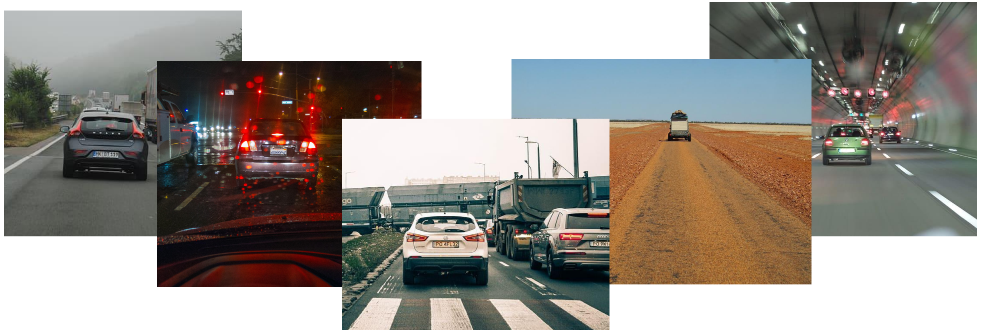
[1]Indian Institute of Technology, Bombay [2]Microsoft Research, India

*ICML 2020, https://arxiv.org/abs/2003.12815, https://github.com/vihari/CSD

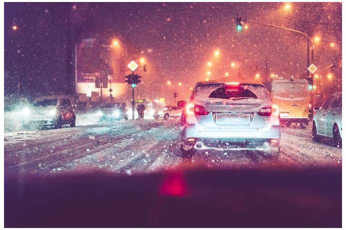
Domain Generalization Problem

Application of self-driving car

Train



Test



Domain Generalization Problem

Automatic Speech Recognition

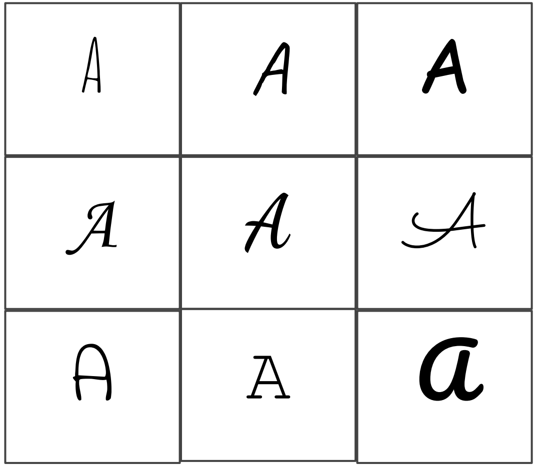
Train

Test

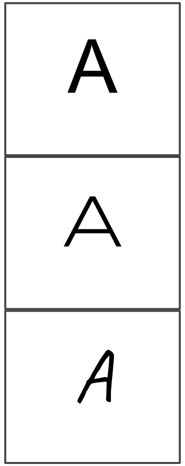
Domain Generalization (DG) Setting

Train on multiple source domains and exploit domain variation during the train time to generalize to new domains.

Exploit multiple train domains during train



Zero-shot transfer to unseen domains



Existing Approaches

- Domain Erasure: Learn domain invariant representations.
- Augmentation: Hallucinate examples from new domains.
- Meta-Learning: Train to generalize on meta-test domains.
- Decomposition: Common-specific parameter decomposition.

Broadly,

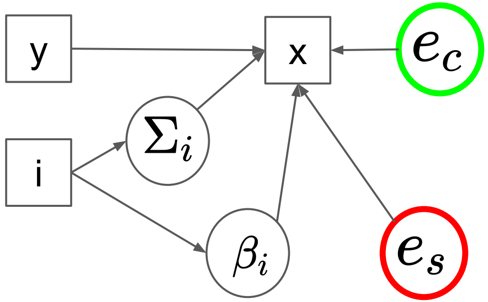
Decomposition < Domain Erasure < Augmentation < Meta-Learning

Contributions

- We provide a principled understanding of existing Domain Generalization (DG) approaches using a simple generative setting.
- We design an algorithm: CSD, that operates on parameter decomposition in to common and specific components. We provide theoretical basis for our design.
- We demonstrate the competence of CSD through an empirical evaluation on a range of tasks including speech. Evaluation and applicability beyond image tasks is somewhat rare in DG.

Simple Linear Classification Setting

Underlying Generative model:



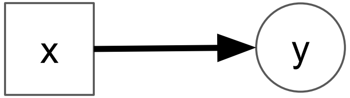
$$x = y(e_c + \beta_i e_s) + \mathcal{N}(0, \Sigma_i)$$

Σ_i , β_i Domain specific noise and scale

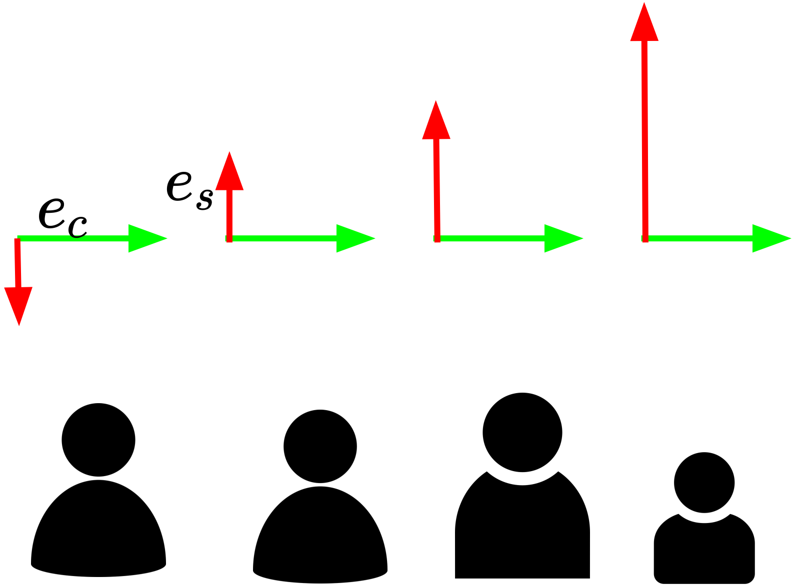
- Coefficient of e_c is constant across domains.
- Coefficient of e_s is domain dependent.

Simple Setting [continued]

Classification task



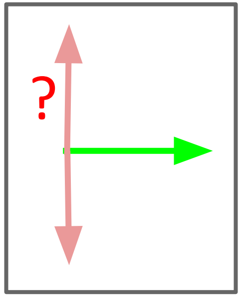
$$x = y(e_c + \beta_i e_s) + \mathcal{N}(0, \Sigma_i)$$



Optimal classifier per domain.

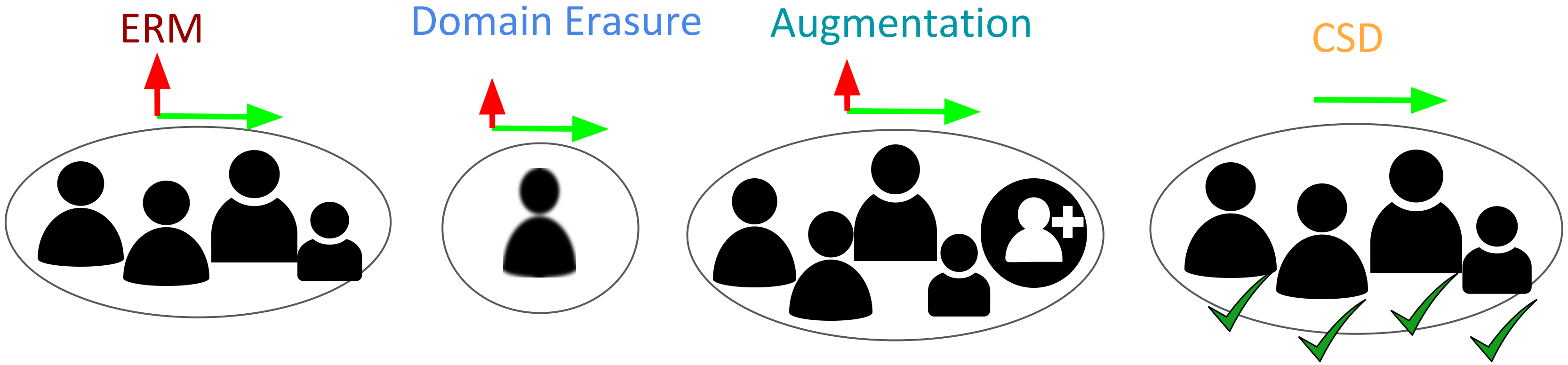
Optimal classifier per domain: $e_c + \beta_i e_s$

For a new domain, cannot predict correlation along e_s



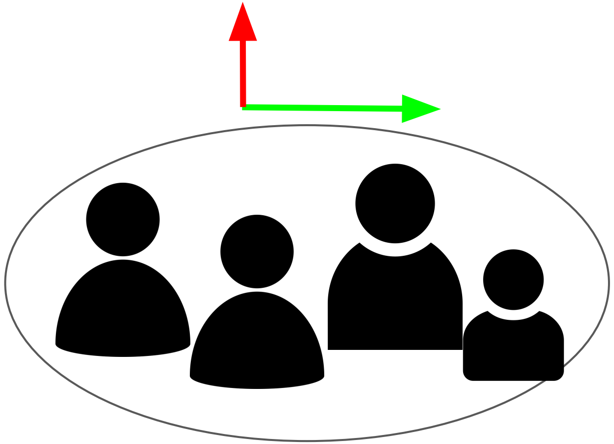
e_c is the generalizing classifier we are looking for!

Evaluation on Simple Setting

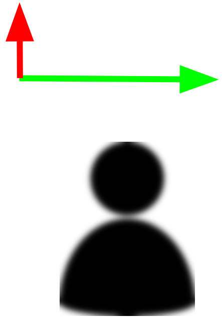


ERM and Domain Erasure

ERM



Domain Erasure

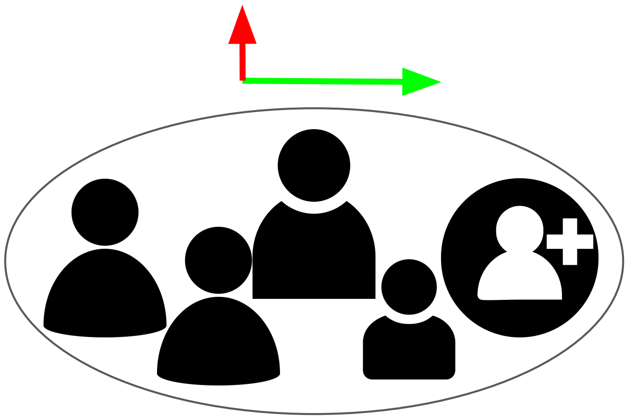


Domain boundaries not considered.
Non-generalizing specific component in solution.

Domain invariant representations.
But all the components carry domain information.

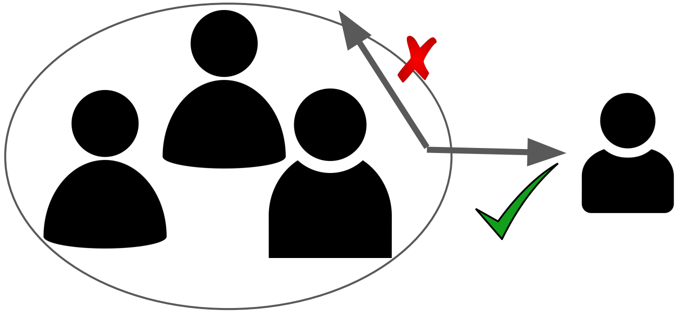
Augmentation and Meta-Learning

Augmentation



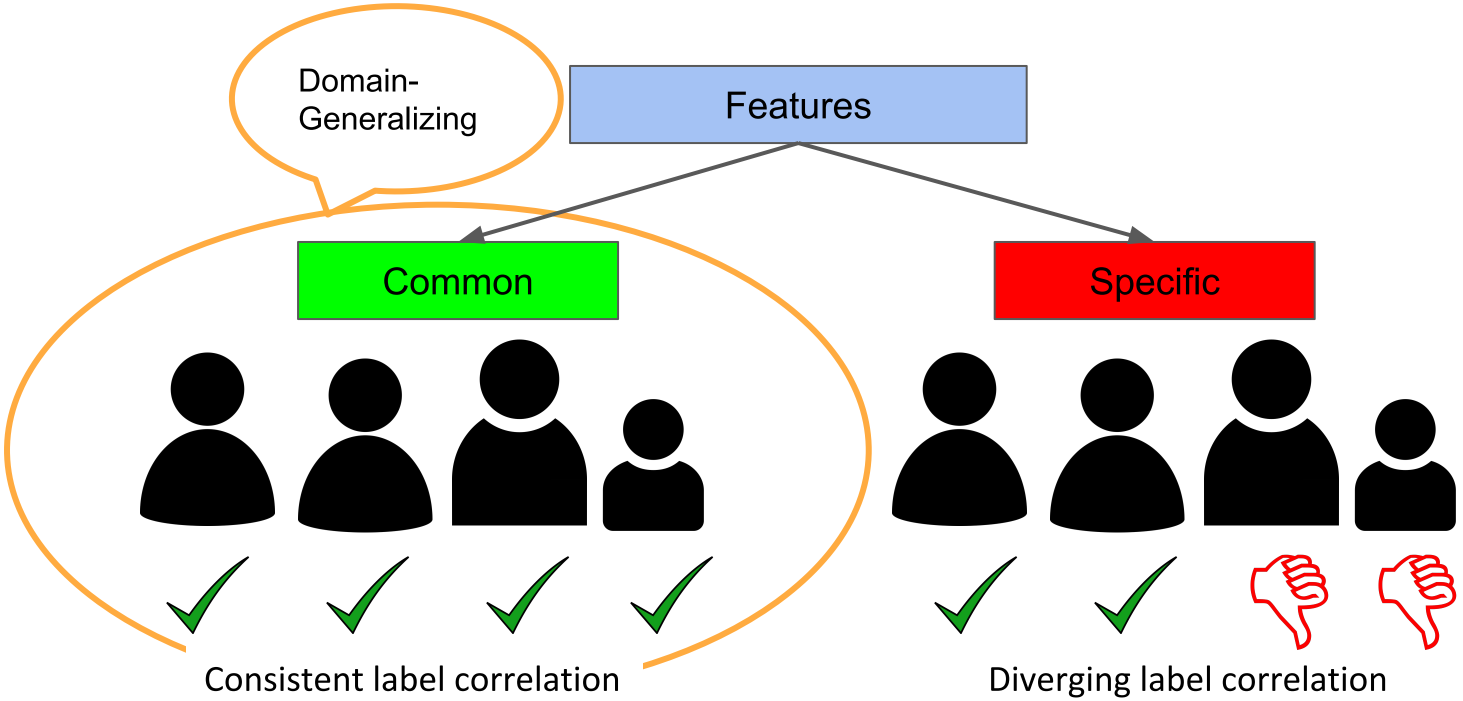
Augments with label consistent examples.
Variance introduced in all the domain-predicting components including common.

Meta-learning



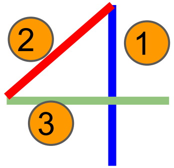
Makes only domain consistent updates.
Could work!
Potentially inefficient when there are large number of domains.

Assumption


Domain-
Generalizing
Features
Common
Specific
Consistent label correlation
Diverging label correlation

Real-world examples of Common-Specific features

Digit recognition with rotation as domain.



Common features:	Specific Features:
• Number of edges: 3	• Angle of 1 = 90 or 90±15.
• Number of corners: 3	• Angle of 2 = 45 or 45±15.
• Angle between 1, 2 or 3	• Angle of 3 = 0 or 0±15.

Domain Generalizing Solution

Desired attribute: A domain generalizing solution should be devoid of any domain specific components.

Our approach:

- Decompose the classifier into common and specific components during train time.
- Retain only common component during test time.

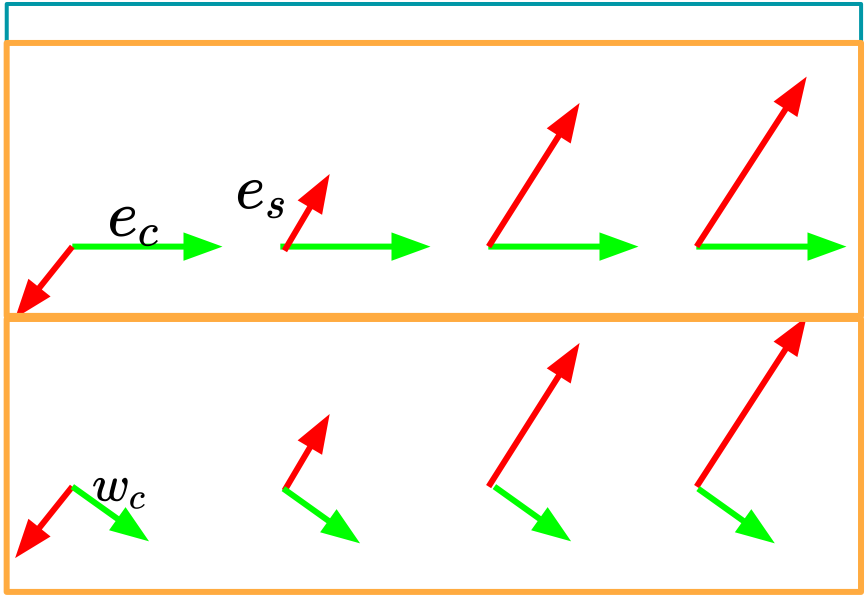
Identifiability Condition

Our decomposition problem is to express optimal classifier of domain *i*: $\tilde{w}_i$ in terms of common and specific parameters: w_c, w_s

$$\tilde{w}_i = w_c + \gamma_i w_s$$

Problem: Several such decompositions.

We are interested in the decomposition where w_c does not have any component of domain variation i.e. $w_c \perp w_s$



In the earlier example, when e_c and e_s are not perpendicular, then $w_c = e_c - P_{e_s} e_c$

Common Specific Decomposition

Let $W := [\tilde{w}_1 \quad \tilde{w}_2 \quad \cdots \quad \tilde{w}_D]$ where $\tilde{w}_i$ is optimal solution for i[th] domain.

Latent dimension of domain space be *k.*

Closed form for common, specific components: $\boldsymbol{w_c} \in \mathbb{R}^m, \boldsymbol{W_s} \in \mathbb{R}^{m \times k}$

Theorem 1. *Given any matrix* $W \in \mathbb{R}^{m \times D}$*, the minimizers of the function* $f(w_c, W_s, \Gamma) = \left\| W - w_c \mathbb{1}^\top - W_s \Gamma^\top \right\|_F^2$*, where* $W_s \in \mathbb{R}^{m \times k}$ *and* $w_c \perp \mathit{Span}\,(W_s)$ *can be computed by the following steps:*

- $w_c \leftarrow \frac{1}{D} W \cdot \mathbb{1}$.
- $W_s, \Gamma \leftarrow \textit{Top-k SVD}\left(W - w_c \mathbb{1}^\top\right)$.
- $w_c^{new} \leftarrow \frac{1}{\left\|(w_c \mathbb{1}^\top + W_s \Gamma^\top)^+ \mathbb{1}\right\|^2} \left(w_c \mathbb{1}^\top + W_s \Gamma^\top\right)^+ \mathbb{1}$.

Number of domain specific components

Optimal solution for domain *i* more generally is: $\tilde{w_i} = w_c + \gamma_i W_s, W_s \in \mathbb{R}^{k \times D}$

How do we pick k? (D is number of train domains)

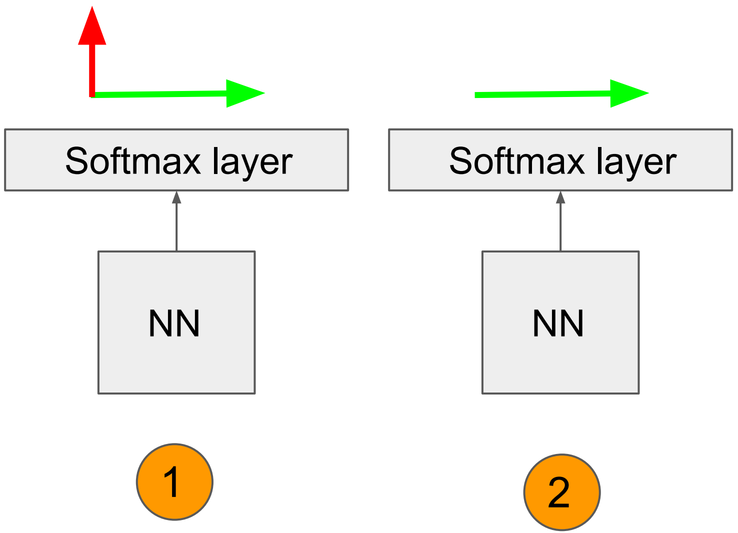
- When k=0, no domain specific component. Same as ERM baseline, **does not generalize**.
- When k=D-1. Common component is effectively free of all domain specific components. However, estimate of W_s can be noisy. Further, the pseudo inverse of W_s in closed form solution makes w_c **estimate unstable** (see theorem 1 of our paper).

Sweet spot for non-zero low value for k.

Extension to deep-net

1. Only final linear layer decomposed.
2. Impose classification loss using common component alone.

So as to encourage representations that do not require specific component for optimal classification.



Common-Specific Low-Rank Decomposition (CSD)

k: latent dimension of domain space
D: Number of domains

(2) Common and Specific softmax parameters
(3) Trainable combination param per domain.

Underlying encoder

Algorithm 1 Common-Specific Low-Rank Decomposition (CSD)

1: **Given:** $D, m, k, C, \lambda, \kappa$,train-data
2: Initialize params $w_c \in \mathbb{R}^{C \times m}, W_s \in \mathbb{R}^{C \times m \times k}$
3: Initialize $\gamma_i \in \mathbb{R}^k : i \in [D]$
4: Initialize params θ of feature network $G_\theta : \mathcal{X} \mapsto \mathbb{R}^m$
5: $\hat{W} = [w_c^T, W_s^T]^T$
6: $\mathcal{R} \leftarrow \sum_{y=1}^{C} \|I_{k+1} - \hat{W}[y]^T \hat{W}[y]\|_F^2$ ▷ Orthonormality constraint
7: **for** $(x, y, i) \in$ train-data **do**
8: $\quad w_i \leftarrow w_c + W_s \gamma_i$
9: $\quad$ loss += $\mathcal{L}(G_\theta(x), y; w_i) + \lambda \mathcal{L}(G_\theta(x), y; w_c)$
10: **end for**
11: Optimize loss+$\kappa\mathcal{R}$ wrt $\theta, w_c, W_s, \gamma_i$
12: **Return** θ, w_c ▷ **for inference**

Common-Specific Low-Rank Decomposition (CSD)

k: latent dimension of domain space
D: Number of domains

(2) Common and Specific softmax parameters
(3) Trainable combination param per domain.

Underlying encoder

Algorithm 1 Common-Specific Low-Rank Decomposition (CSD)

1: **Given:** $D, m, k, C, \lambda, \kappa$,train-data
2: Initialize params $w_c \in \mathbb{R}^{C\times m}, W_s \in \mathbb{R}^{C\times m\times k}$
3: Initialize $\gamma_i \in \mathbb{R}^k : i \in [D]$
4: Initialize params θ of feature network $G_\theta : \mathcal{X} \mapsto \mathbb{R}^m$
5: $\hat{W} = [w_c^T, W_s^T]^T$
6: $\mathcal{R} \leftarrow \sum_{y=1}^{C} \|I_{k+1} - \hat{W}[y]^T \hat{W}[y]\|_F^2$ ▷ Orthonormality constraint
7: **for** $(x, y, i) \in$ train-data **do**
8: $w_i \leftarrow w_c + W_s\gamma_i$
9: loss $+= \mathcal{L}(G_\theta(x), y; w_i) + \lambda\mathcal{L}(G_\theta(x), y; w_c)$
10: **end for**
11: Optimize loss$+\kappa\mathcal{R}$ wrt $\theta, w_c, W_s, \gamma_i$
12: **Return** θ, w_c ▷ **for inference**

Common-Specific Decomposition (CSD)

k: number of specific components

Initialize common, specific classifiers and a domain-specific combination weights.

Common classifier should be orthogonal to the span of specific classifiers (identifiability constraint)

Classification loss using common classifier only and specialized classifiers

Algorithm 1 Common-Specific Low-Rank Decomposition (CSD)

Given: k, $\bigcup\{x, y, i\}$, encoder G_θ

Initialize $w_c, W_s, \gamma_i \in \mathbb{R}^k$

$\mathcal{R} \leftarrow$ Orthonormal Loss($[w_c, W_s]$)

$$L \leftarrow \sum_{x,y,i} \text{Loss}(G_\theta(x),\ y;\ \theta,\ w_c) + \text{Loss}(G_\theta(x),\ y;\ \theta,\ w_c + \gamma_i W_s)$$

Optimize L+$\mathcal{R}$

Return w_c ▷ For inference

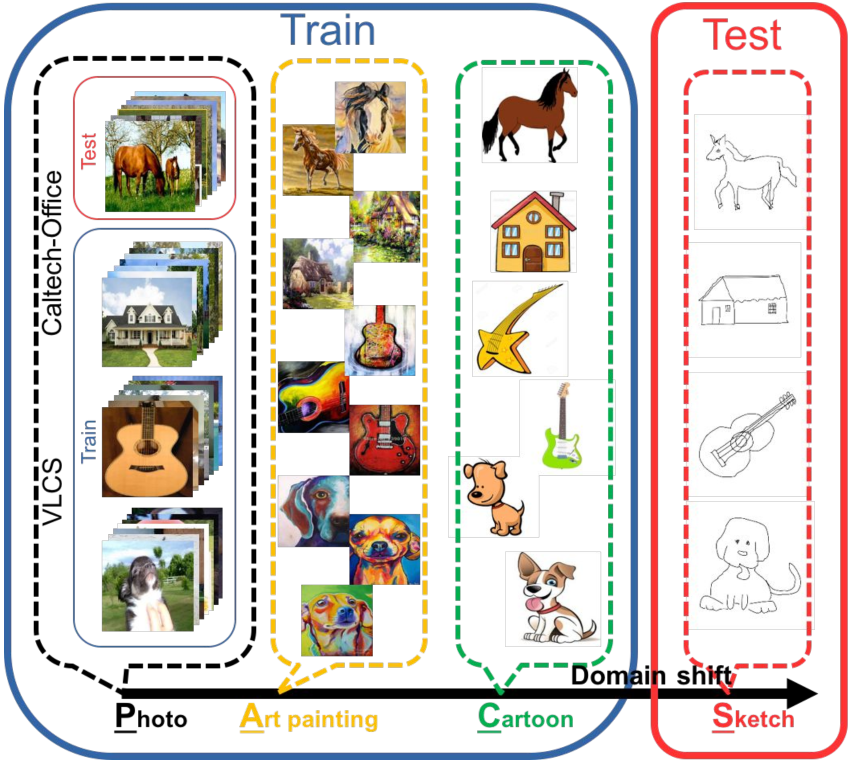
Retain only the the generalizing common classifier.

Results

Evaluation

Evaluation scores for DG systems is the classification accuracy on the unseen and potentially far test domains.

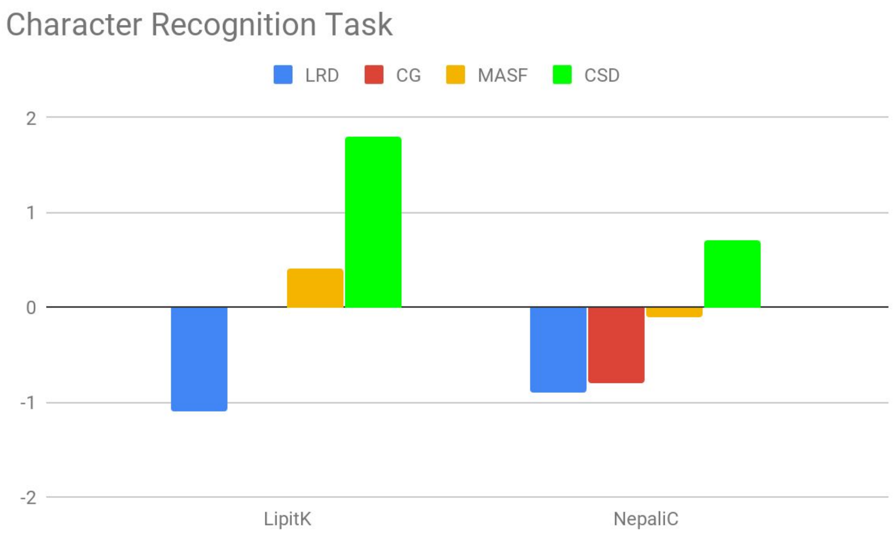
Setting for PACS dataset shown to the right.



PACS dataset. Source: [PACS]

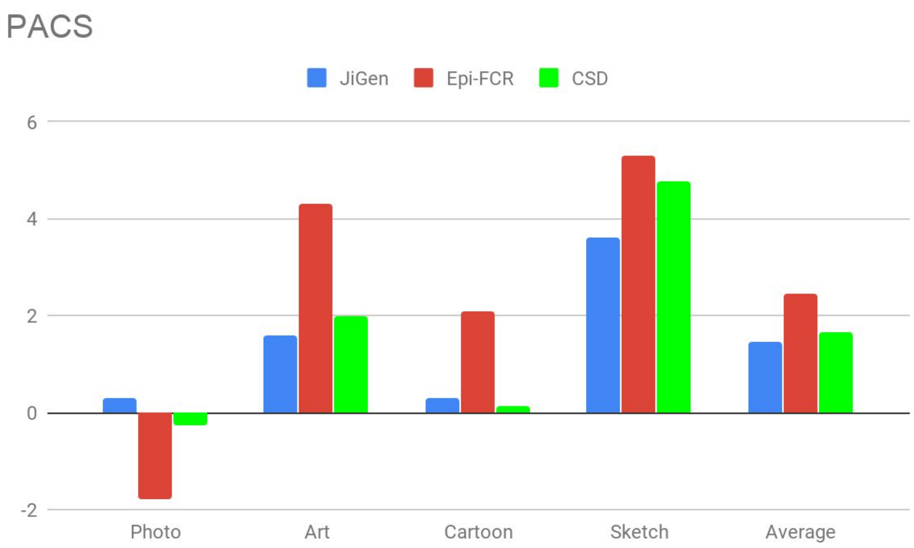
Image tasks

- LipitK and NepaliC are handwritten character recognition tasks.
- Shown are the accuracy gains over the ERM baseline.
- LRD, CG, MASF are strong contemporary baselines.
- CSD consistently outperforms others.



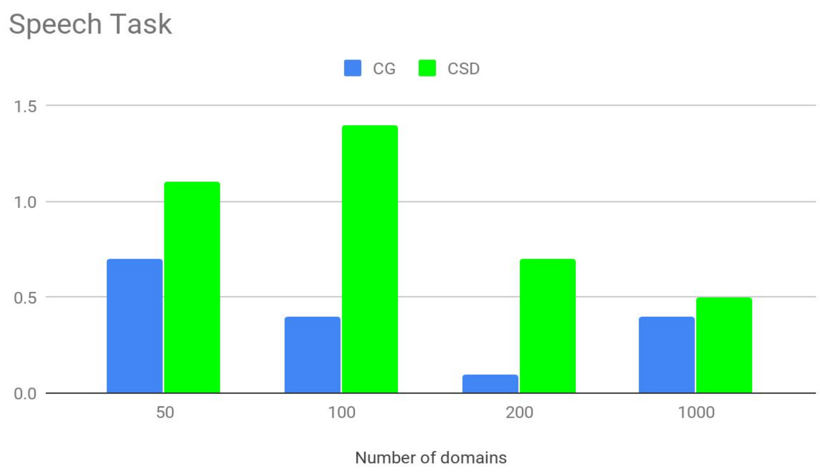
PACS

- Photo-Art-Cartoon-Sketch (PACS) is a popular benchmark for Domain Generalization.
- Shown are the relative classification accuracy gains over baseline.
- JiGen and Epi-FCR are latest strong baselines.
- CSD despite being simple is competitive.



Speech Tasks

- Improvement over baseline on speech task for varying number of domains, shown on X-axis.
- CSD is consistently better.
- Decreasing gains over baseline as number of train domains increase.



Implementation and Code

- Our code and datasets are publicly available at https://github.com/vihari/csd.
- In strong contrast to typical DG solutions, our method is extremely simple and has a runtime of only x1.1 of ERM baseline.
- Since our method only swaps the final linear layer, it could be easier to incorporate in to your code-stack.
- We encourage you to try CSD if you are working on a Domain Generalization problem.

Conclusion

- We considered a natural multi-domain setting and showed how existing solutions could still overfit on domain signals.
- Our proposed algorithm: CSD effectively decomposes classifier parameters into a common and a low-rank domain-specific part. We presented analysis for identifiability and motivated low-rank assumption for decomposition.
- We empirically evaluated CSD against six existing algorithms on six datasets spanning speech and images and a large range of number of domains. We show that CSD is competent and is considerably faster than existing algorithms, while being very simple to implement.