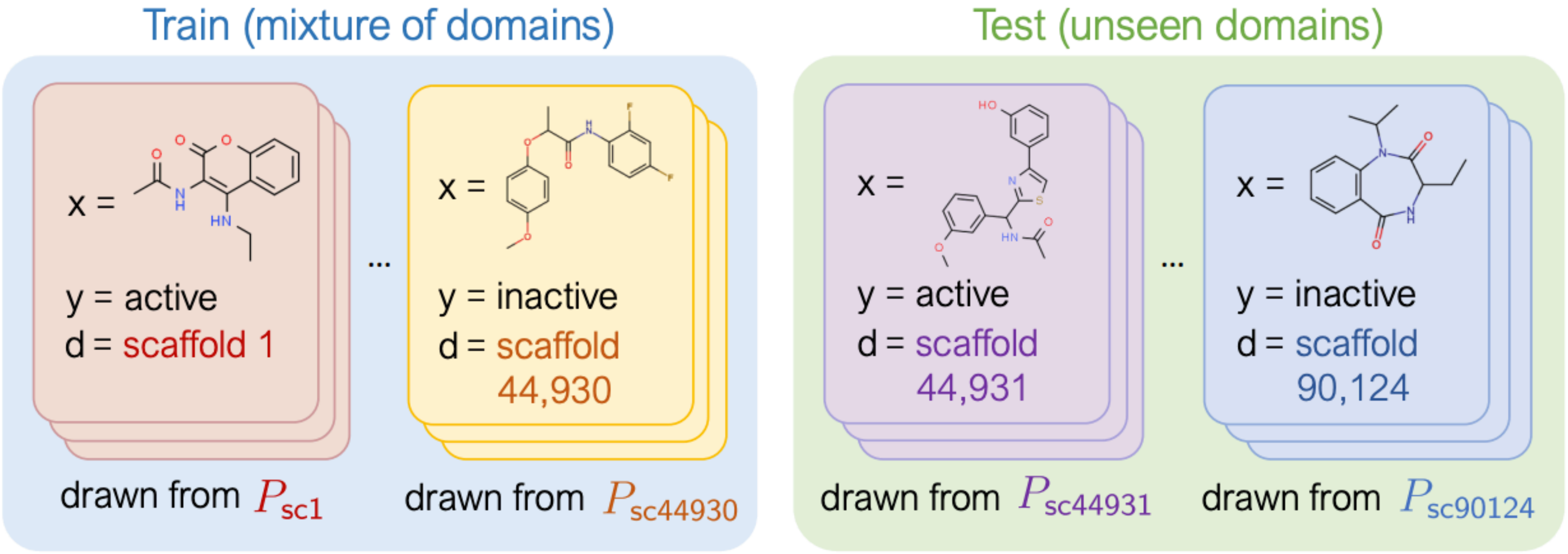
Project and **Pro**be: Sample-Efficient Domain Adaptation by Interpolating Orthogonal Features

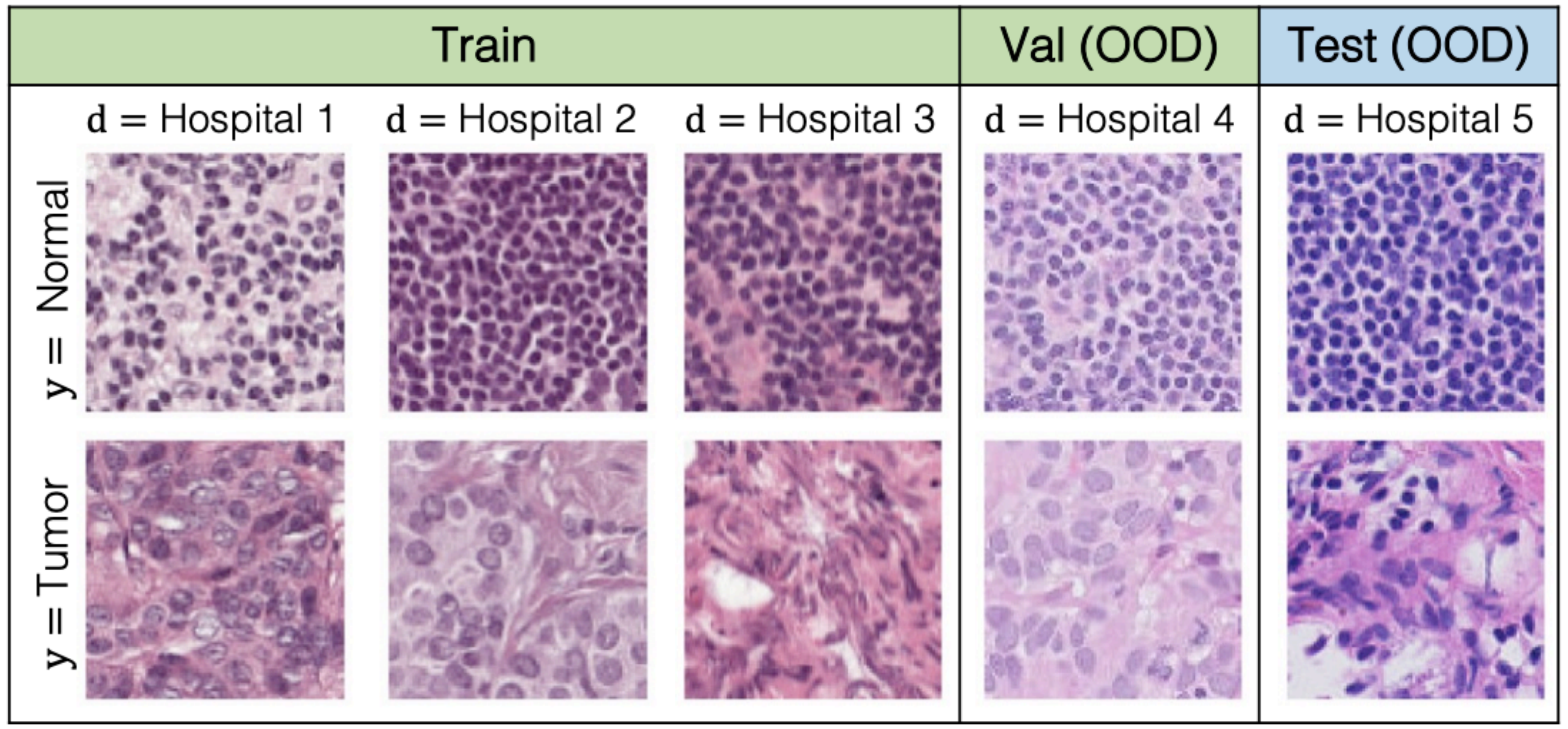
Annie S. Chen*[1], Yoonho Lee*[1], Amrith Setlur[2], Sergey Levine[3], Chelsea Finn[1]
Stanford University[1], Carnegie Mellon University[2], UC Berkeley[3]

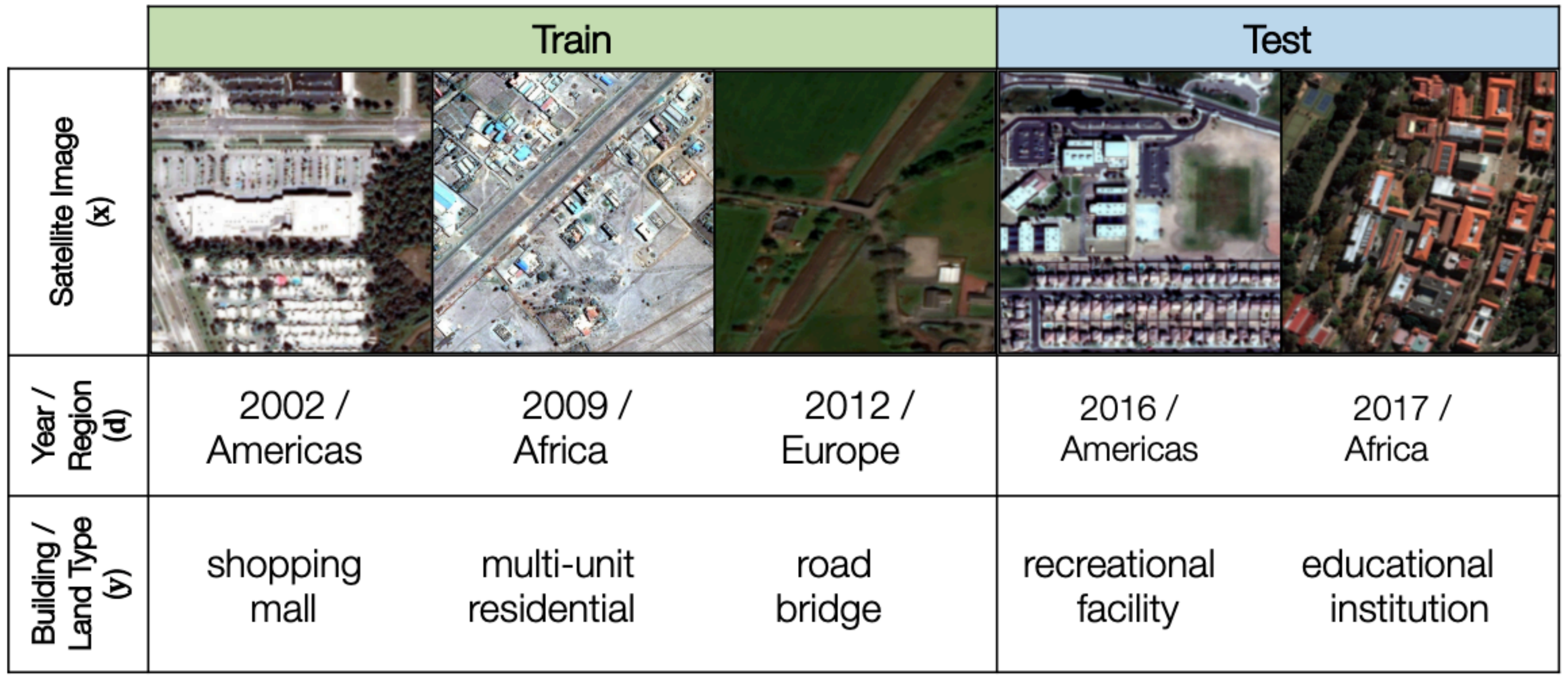
Deep Learning: Classics and Trends, ML Collective

March 17, 2023

Distribution Shifts







Figures from "Wilds: A benchmark of in-the-wild distribution shifts" (2021)

Transfer Learning for Adaptation

A reliable way of adapting to distribution shifts: leverage a small amount of labeled data from the new ***target domain***.

Problem setting:

Backbone pre-trained on generic large dataset (optional)

Large dataset from relevant *source domain*

Small dataset from *target domain*

Approaches to Adaptation

Less inductive bias, more flexible

Approaches to Adaptation

ERM
w/ Source

Approaches to Adaptation

ERM
w/ Source

+ Reusable
- Inflexible
- Susceptible to shortcuts

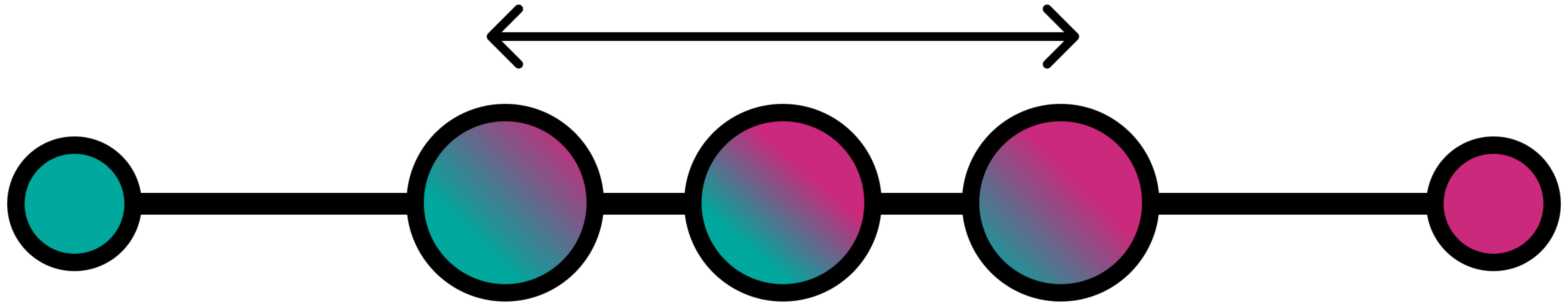
Fine-Tuning
w/ Target

+ Adaptive
- May overfit

Approaches to Adaptation

+Sample-efficient?

+Adaptive?



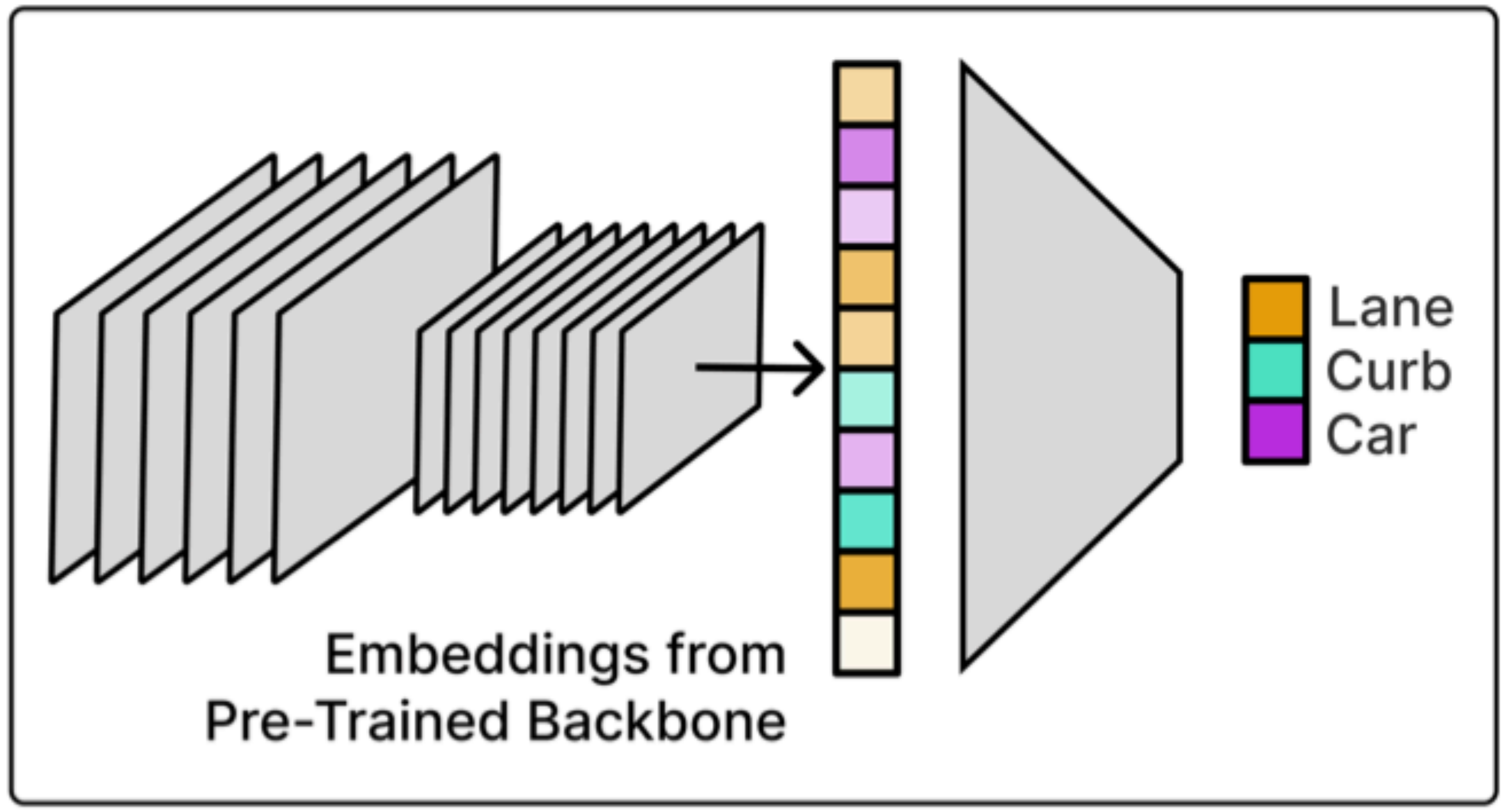
ERM
w/ Source

+ Reusable
- Inflexible
- Susceptible to shortcuts

Fine-Tuning
w/ Target

+ Adaptive
- May overfit

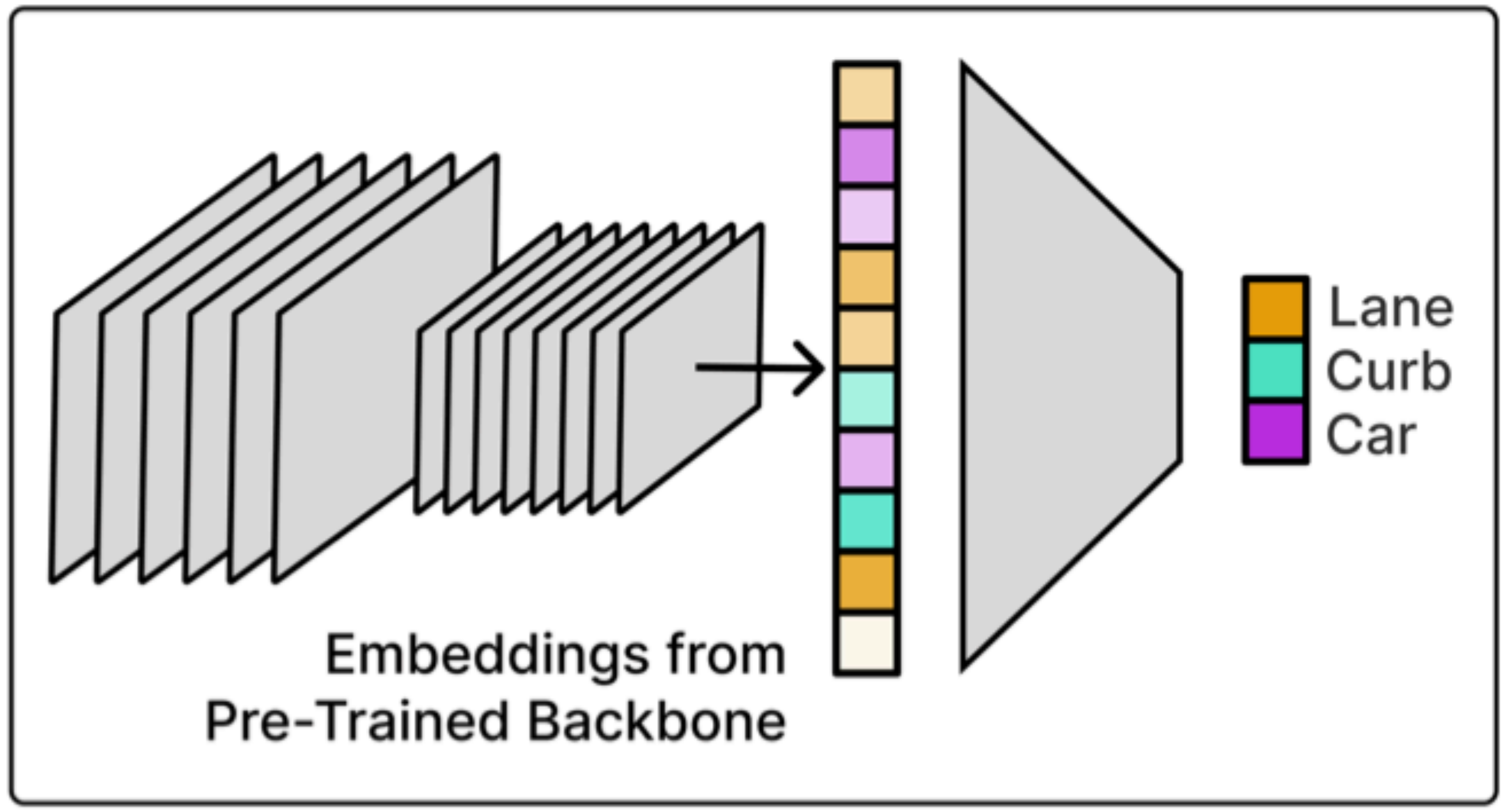
Project and Probe



(a) **Project** with Large Source Dataset

(b) **Probe** with Small Target Dataset

Step 1: **Project** with Source Data



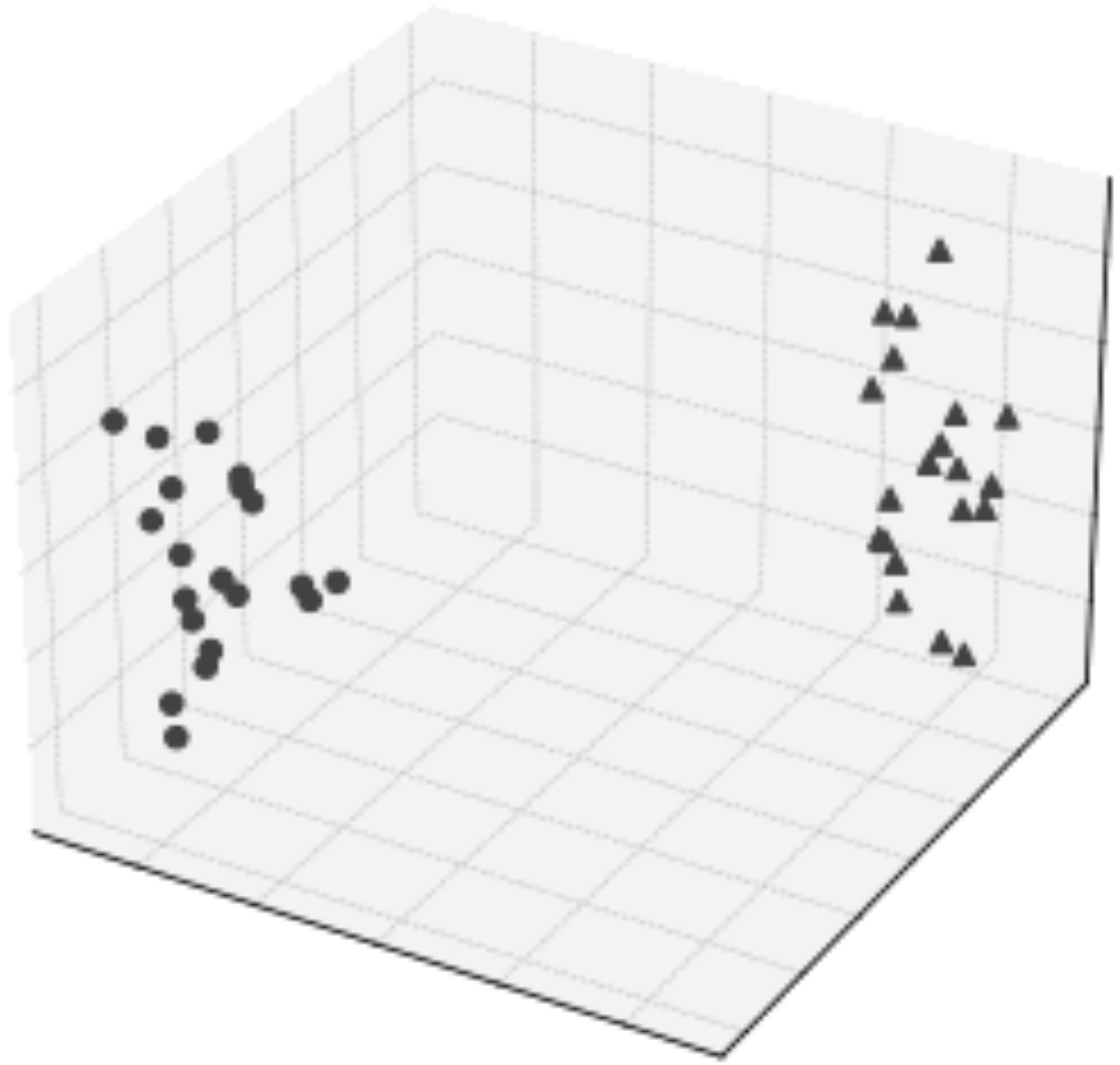
(a) **Project** with Large Source Dataset

For binary classification:

- Initialize d classifiers (D x d matrix)
- Train each classifier with:
  - Cross-entropy loss on source data
  - Orthogonality constraint w.r.t. all previous

$$\Pi_i = \arg\min \mathbb{E}_{(x,y)\sim\mathcal{D}_S}\mathcal{L}(\Pi_i(f(x)), y) \quad \text{s.t.} \quad \Pi_j \perp \Pi_i \text{ for all } j < i.$$

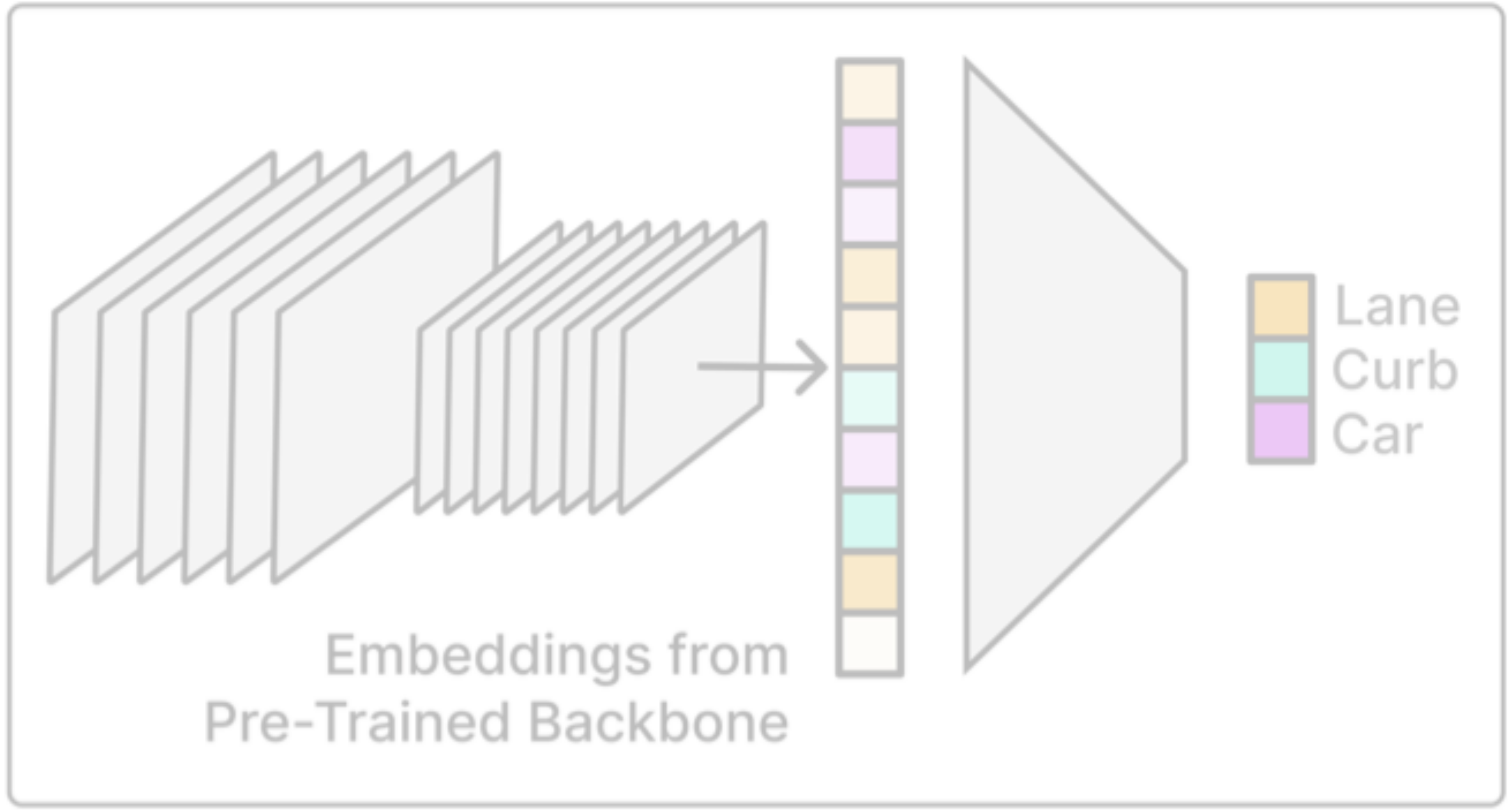
Step 1: **Project** with Source Data



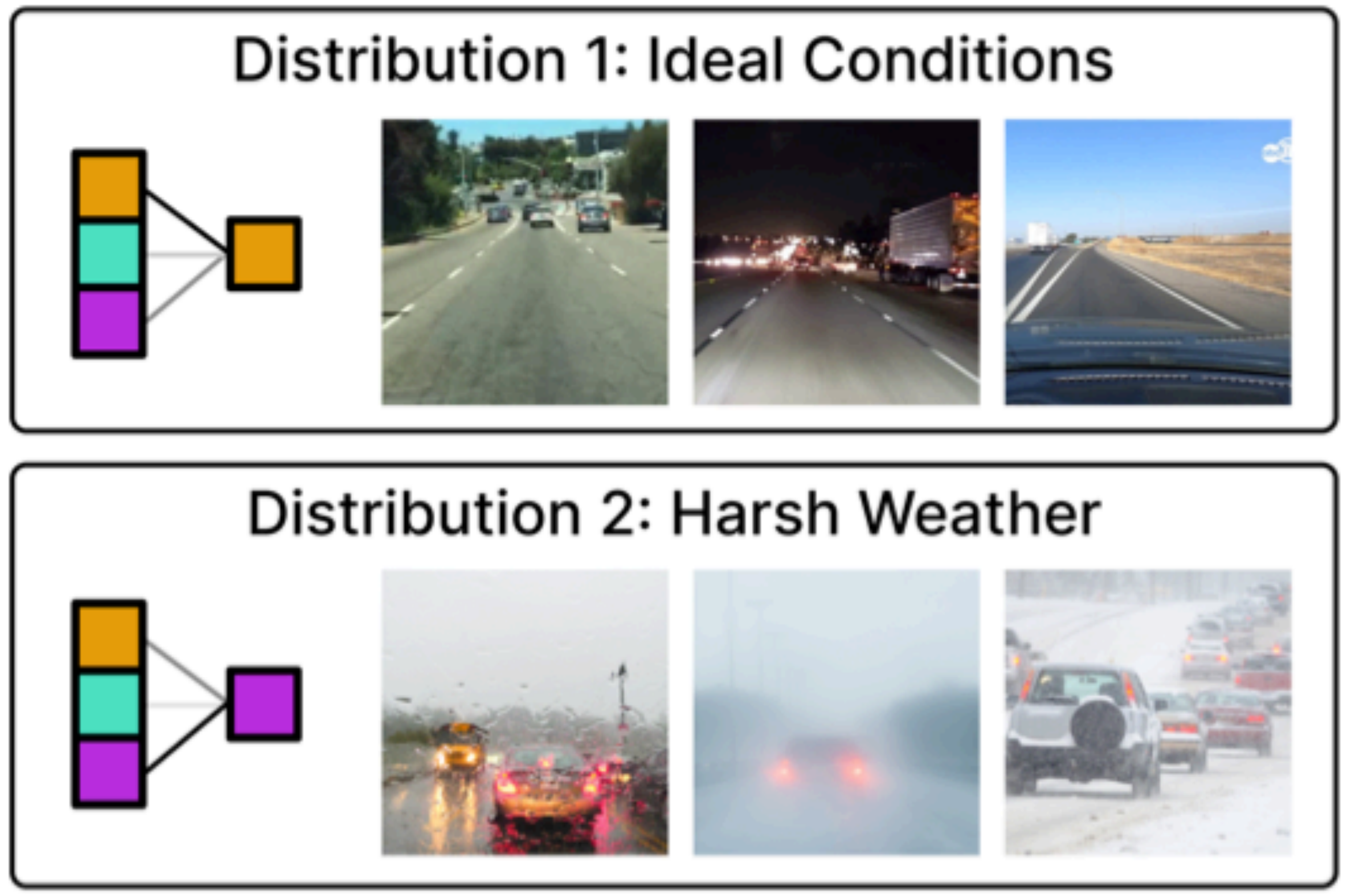
```python
def learn_feature_space_basis(x, y, num_features):
    projection = torch.nn.Linear(x.shape[1], num_features)
    opt = torch.optim.AdamW(projection.parameters(), lr=0.01, weight_decay=0.01)
    max_steps = 100
    for i in range(max_steps):
        logits = projection(x)
        loss = F.binary_cross_entropy_with_logits(logits, y, reduction="none").mean()
        opt.zero_grad()
        loss.backward()
        opt.step()
        # Enforce orthogonality; we're performing projected gradient descent
        Q, R = torch.linalg.qr(linear_model.weight.detach().T)
        projection.weight.data = (Q * torch.diag(R)).T
    feature_space = projection.weight.detach().T
    return feature_space
```

Simple: 15 lines of PyTorch code!

Step 2: **Probe** with Target Data



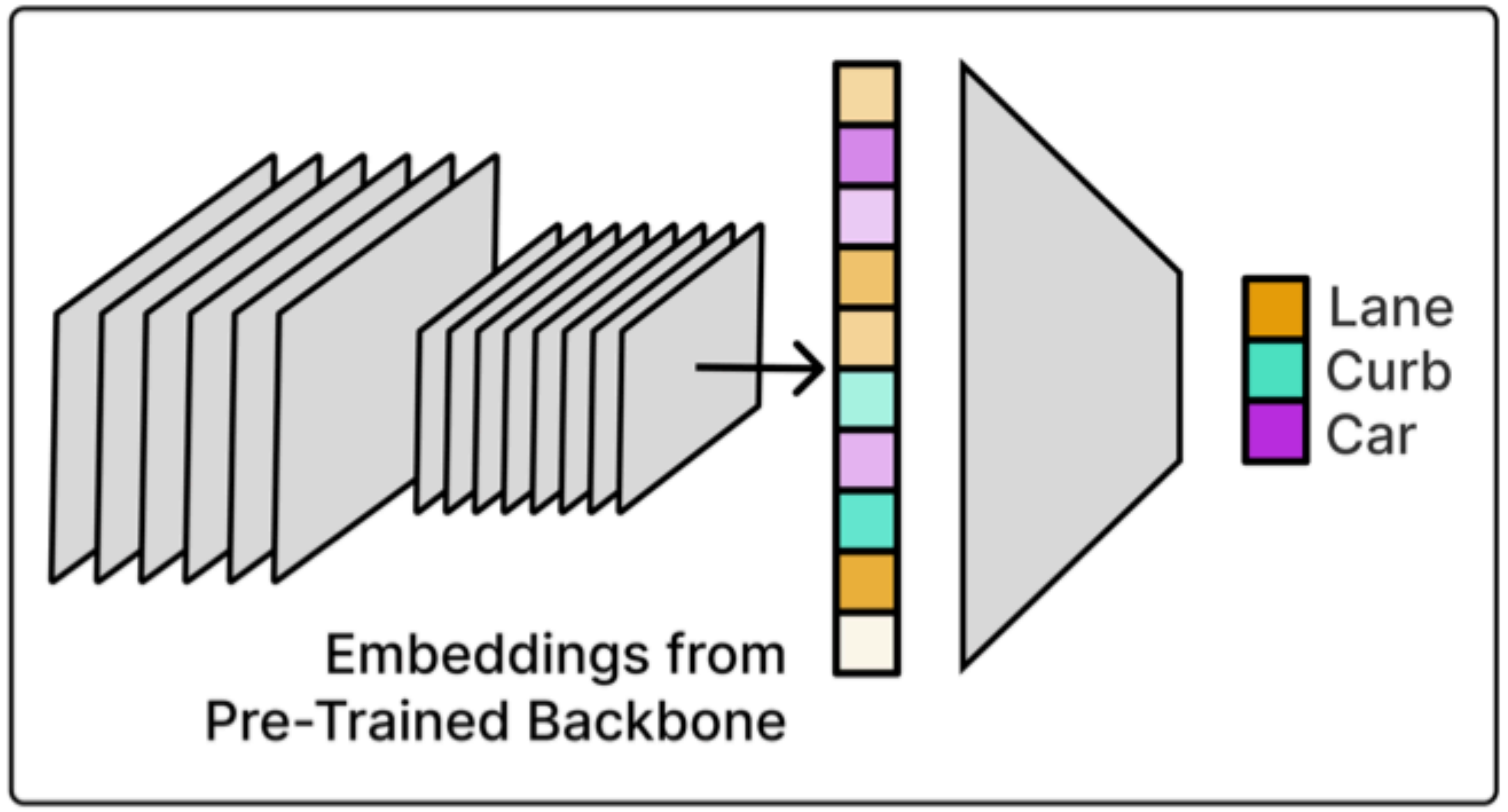
(a) **Project** with Large Source Dataset



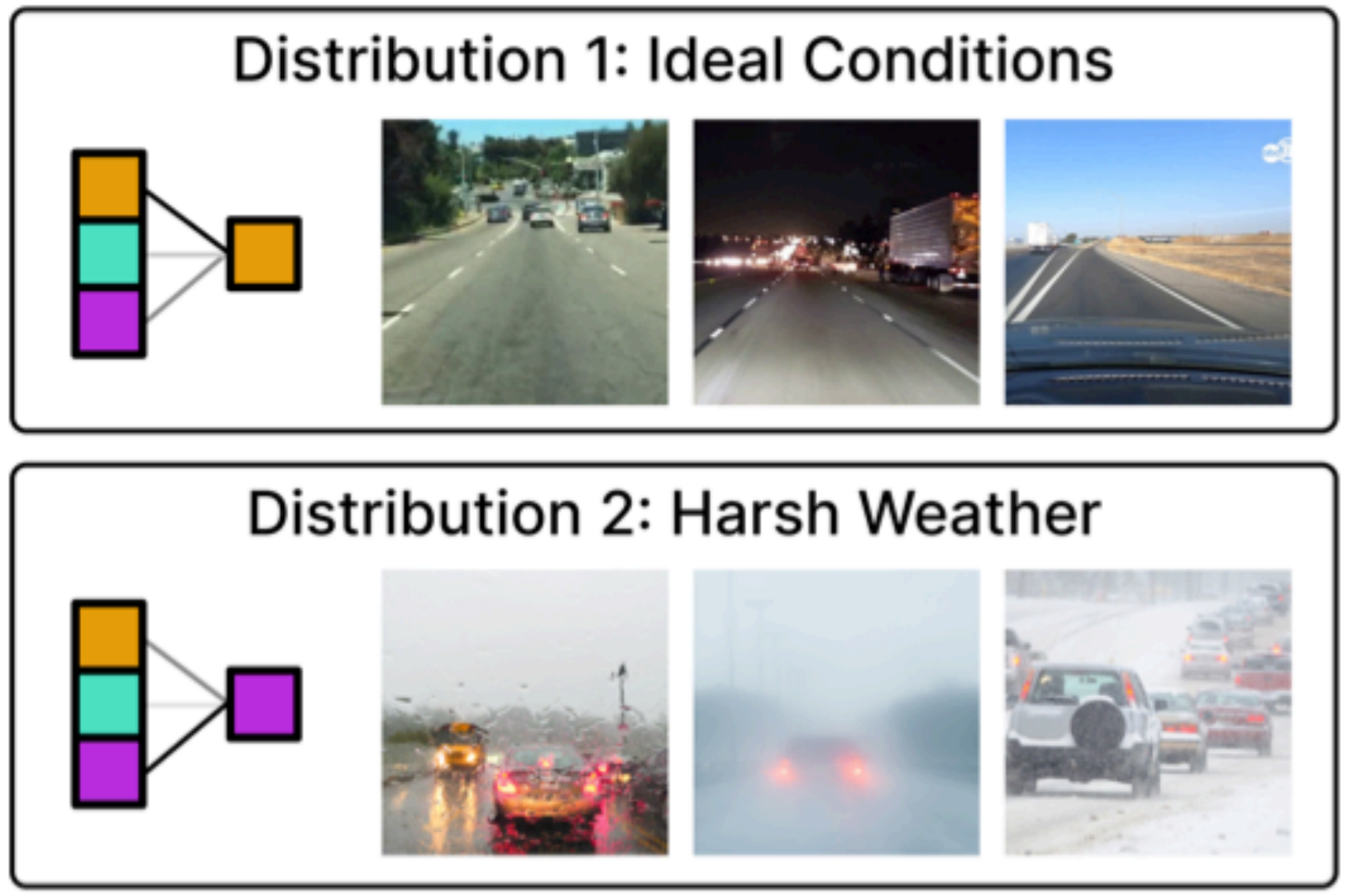
(b) **Probe with Small Target Dataset**

$$\arg\min \mathbb{E}_{(x,y)\sim\mathcal{D}_T}\mathcal{L}(g(\Pi(f(x))),y).$$

Project and Probe (Pro²)



(a) **Project with Large Source Dataset**



(b) **Probe with Small Target Dataset**

Project: Learn linear projection of pre-trained embeddings onto orthogonal directions

Probe: Interpolate between projected features w/ a small target dataset

+ Very lightweight: 30,000 experiments in <24 hrs, on CPUs only!

Pro² induces a favorable bias-variance tradeoff

Theorem 3 (bias-variance tradeoff). *When the conditions in Lemma 2 hold and when $\|\mathbf{x}\|_\infty = \mathcal{O}(1)$, for B-bounded loss l, w.h.p. $1-\delta$, the excess risk for the solution $\hat{\mathbf{w}}_d$ of* Pro² *that uses d features is $\mathcal{L}_T(\hat{\mathbf{w}}_d) - \min_{\mathbf{w}\in\mathcal{W}} \mathcal{L}_T(\mathbf{w})$*

$$\lesssim \|(\boldsymbol{I}_D - \boldsymbol{\Pi}_d)\mathbf{w}_T^*\|_2 + \left(\frac{\sqrt{d} + B\sqrt{\log(1/\delta)}}{\sqrt{M}}\right), \tag{2}$$

where the first term controls the bias and the second controls the variance.

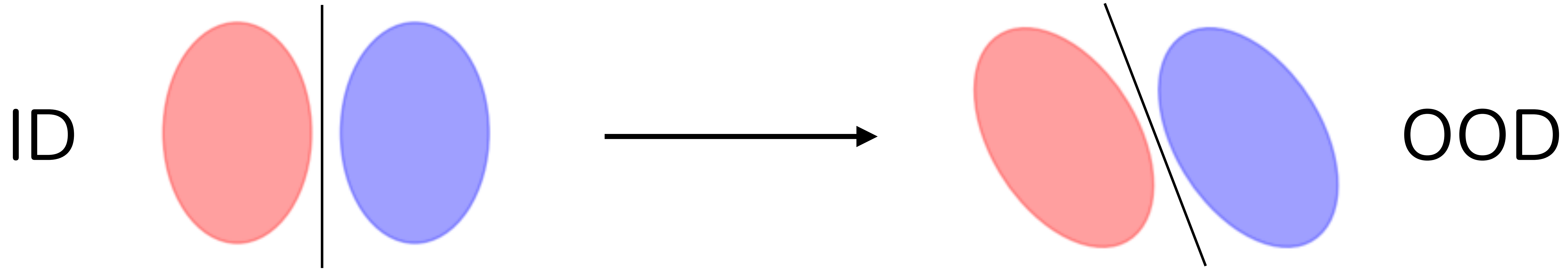
A small dataset entails high variance.

We can reduce variance with a low-dimensional projection.

The projection introduces additional bias, which is low when the most important directions are covered (possible when the shift is not too severe).

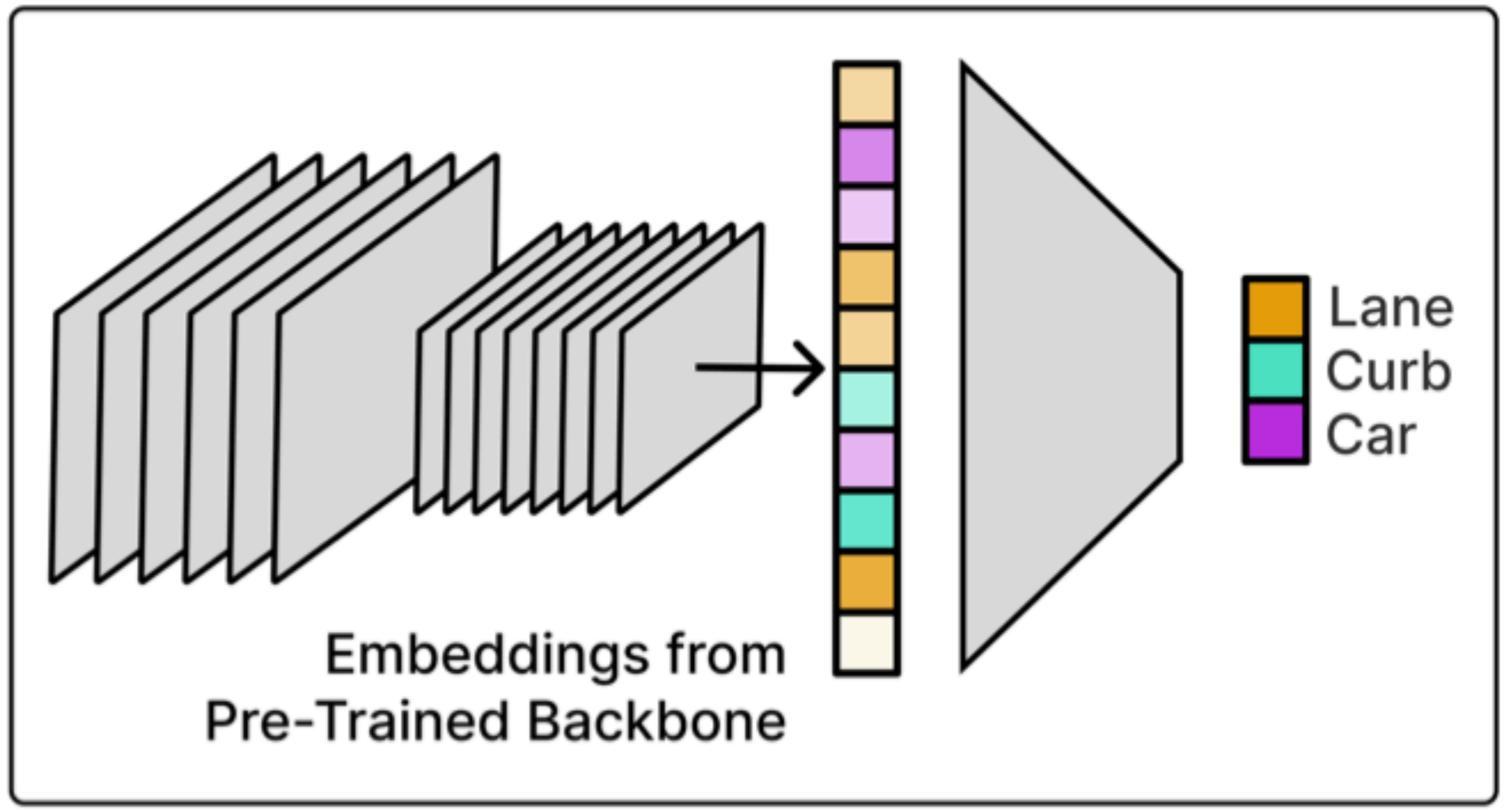
Pro² recovers important & diverse features to reduce bias.

Bias-variance in shifted Gaussian model

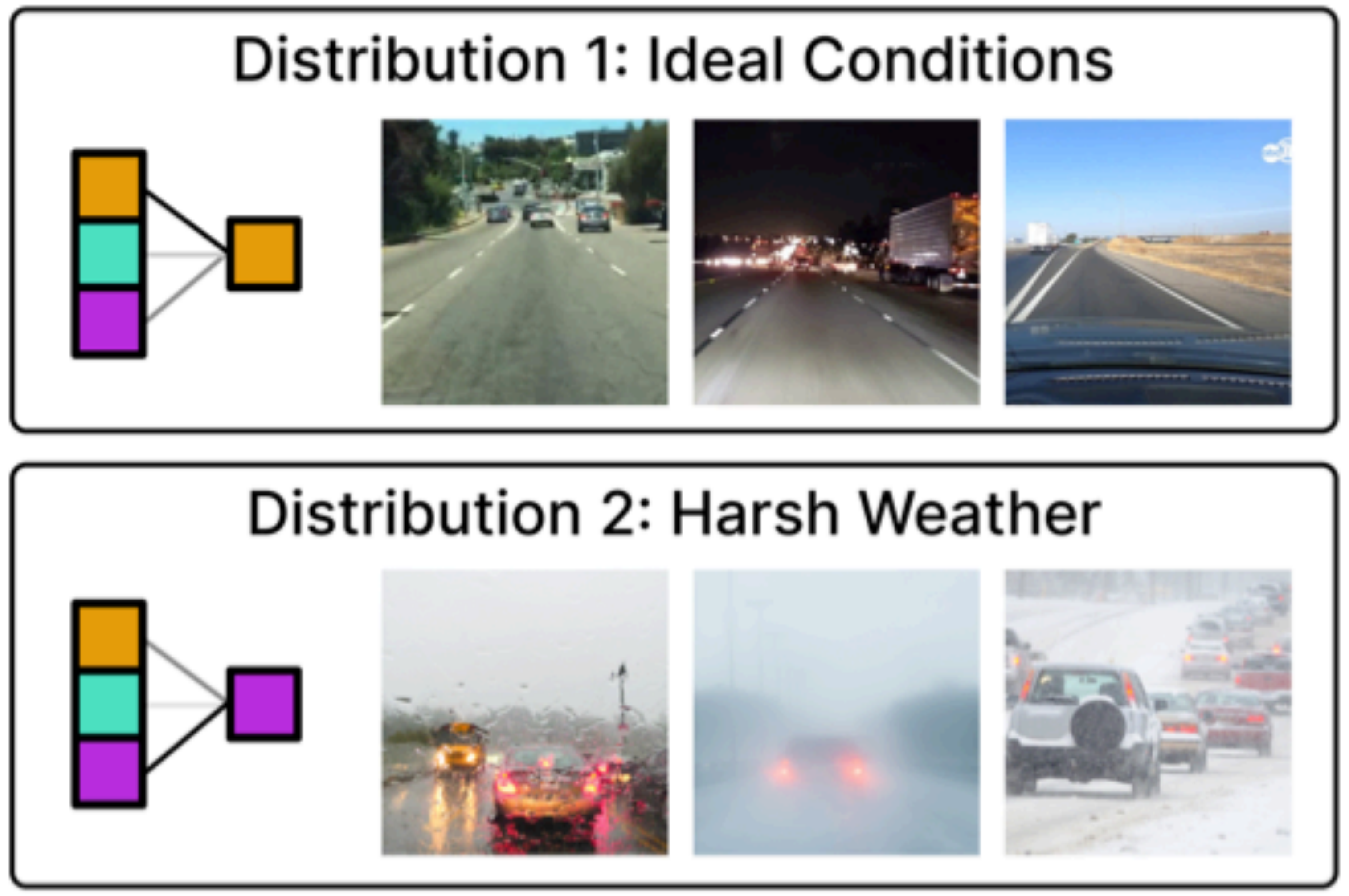


For a small dataset size, bias & variance can be balanced using a smaller projection dim if important directions are covered.

Project and Probe (Pro²)



(a) **Project** with Large Source Dataset



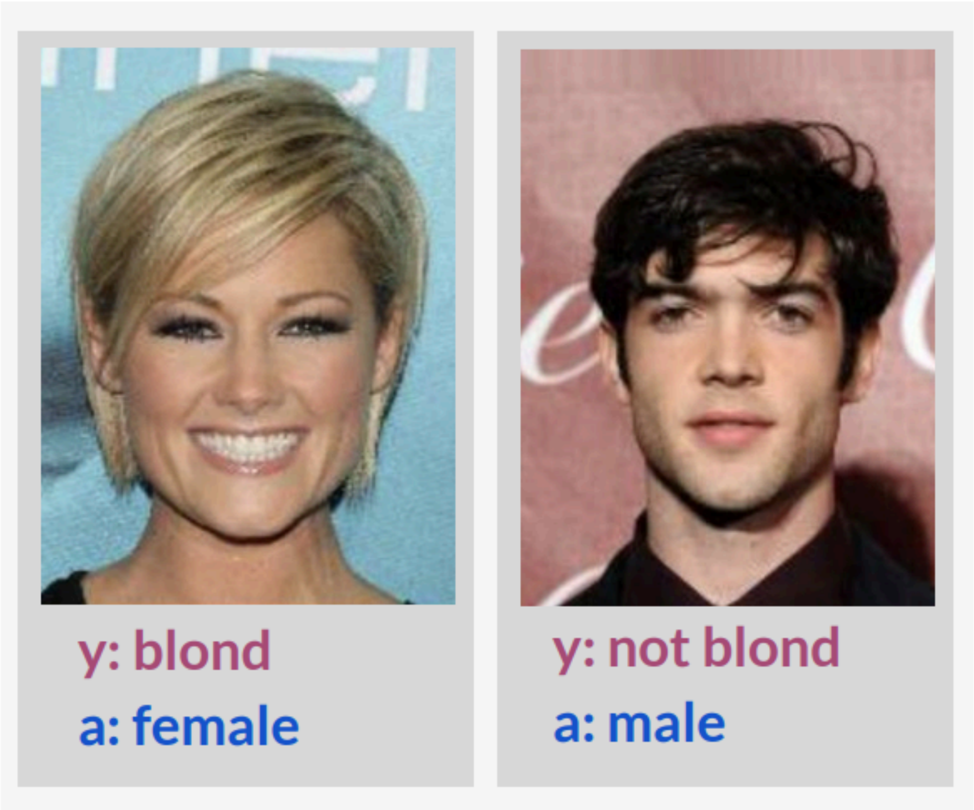
(b) **Probe** with Small Target Dataset

1) Project: Learn linear projection of pre-trained embeddings onto orthogonal directions
2) Probe: Interpolate b/w projected features w/ a small amt of target data

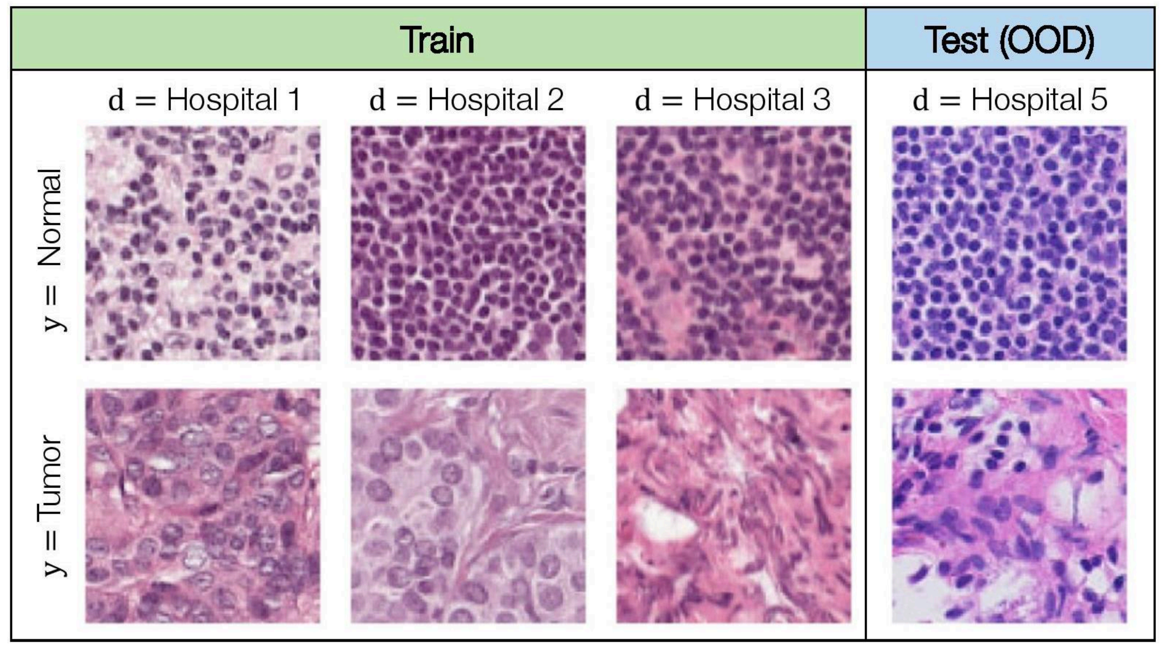
\+ Adaptive b/c projection learns diverse (orthogonal) features
\+ Efficient b/c projection learns useful (predictive) features

Experiments

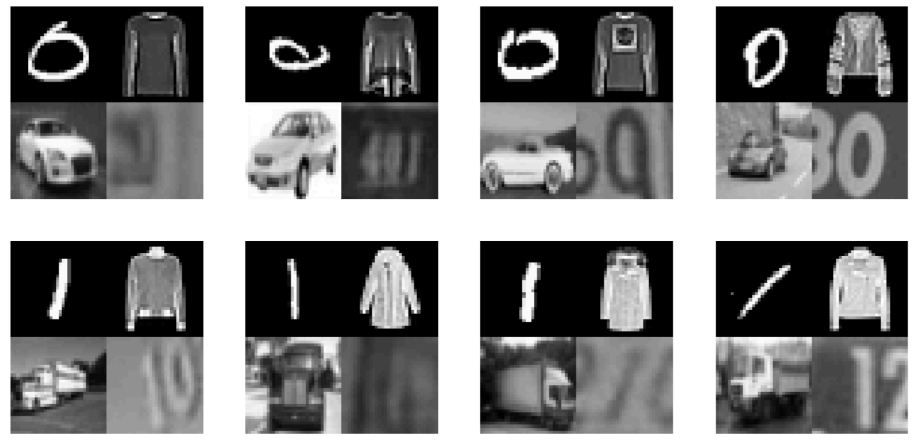
Waterbirds



CelebA



Camelyon17



4-way Collages

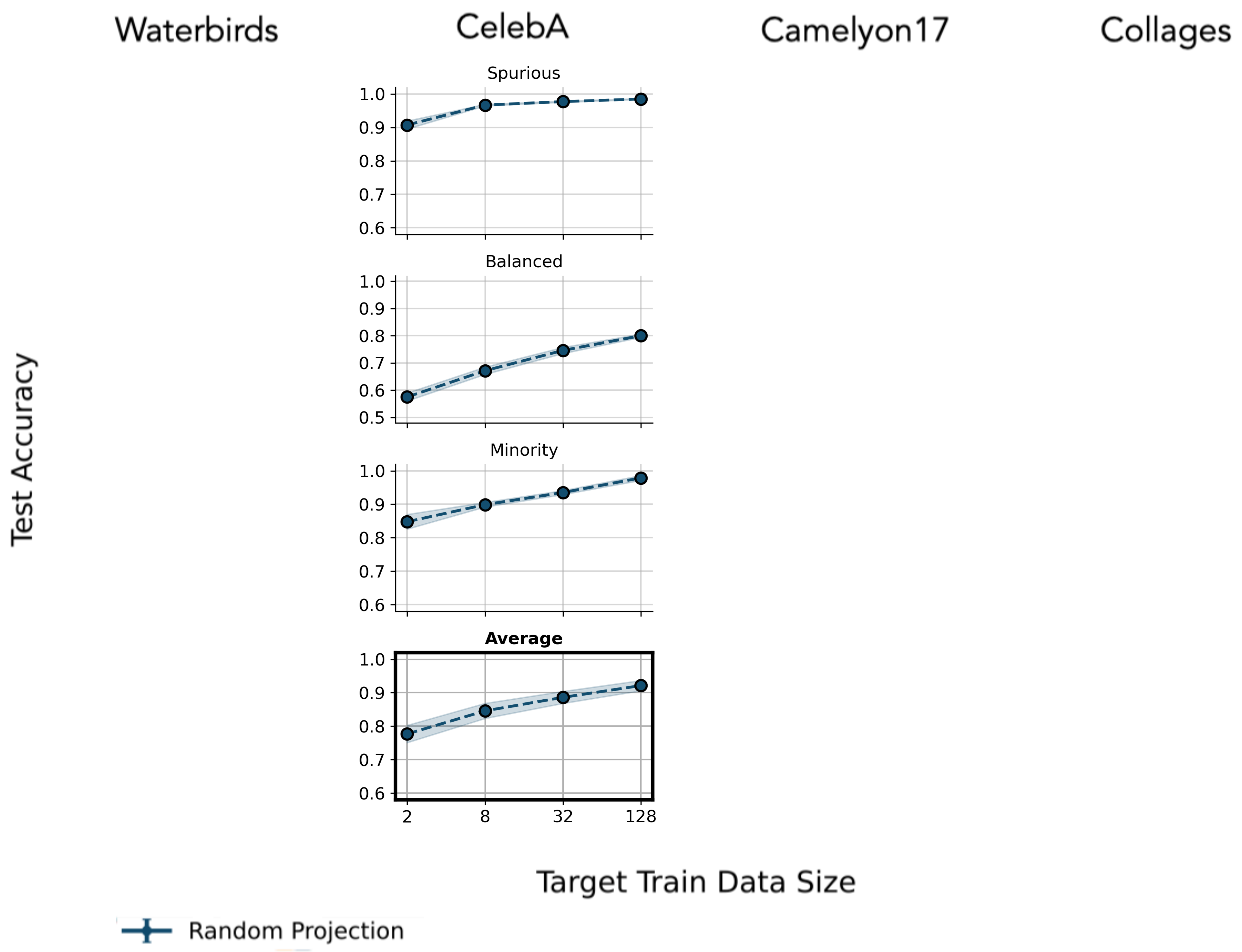
Comparisons

Random Projection: Project onto random orthogonal features

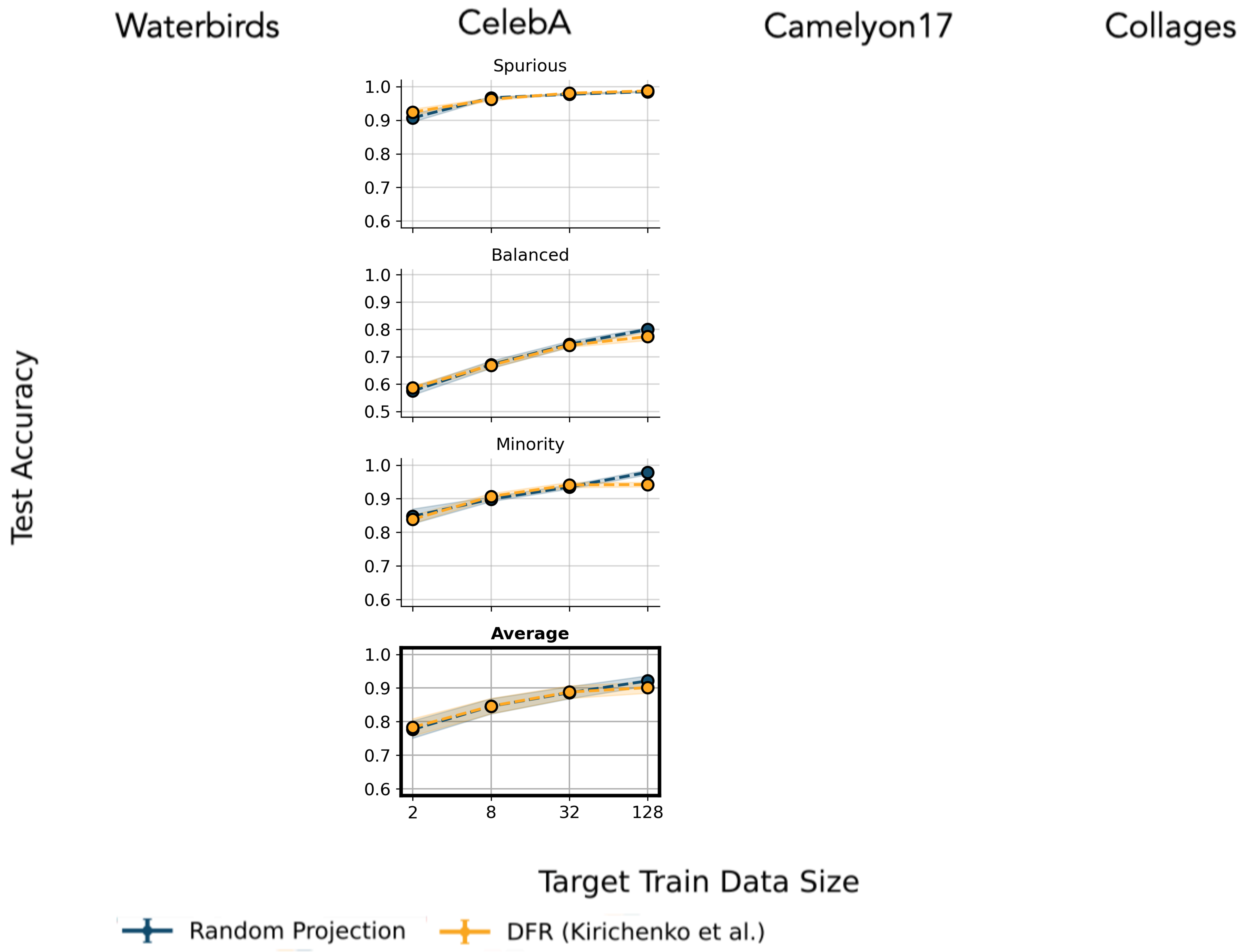
DFR (Kirichenko et al. 2022): standard linear probing on *target data*

Teney et al. 2022: minimize alignment of input gradients over pairs of features

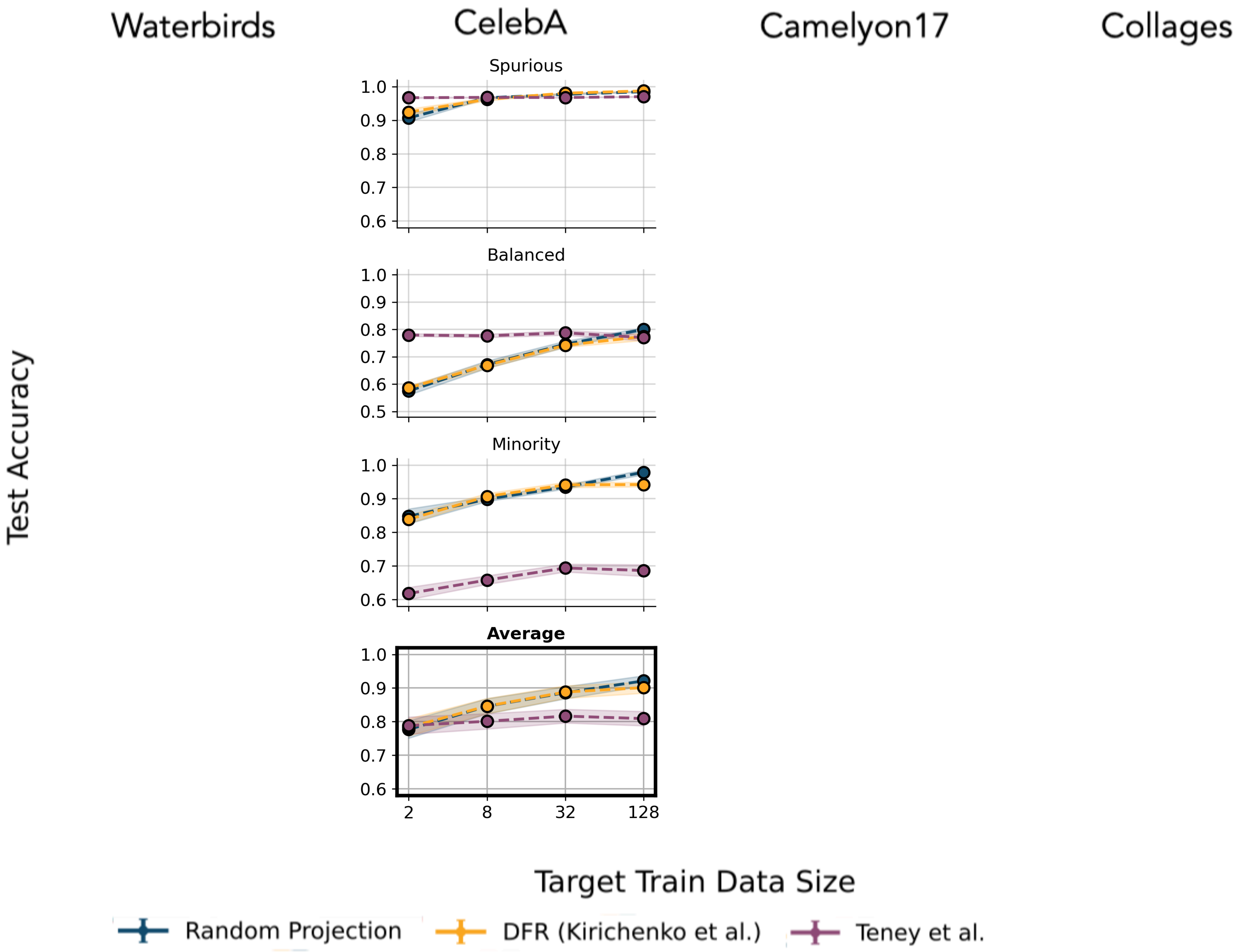
Pro² results in higher adaptation acc



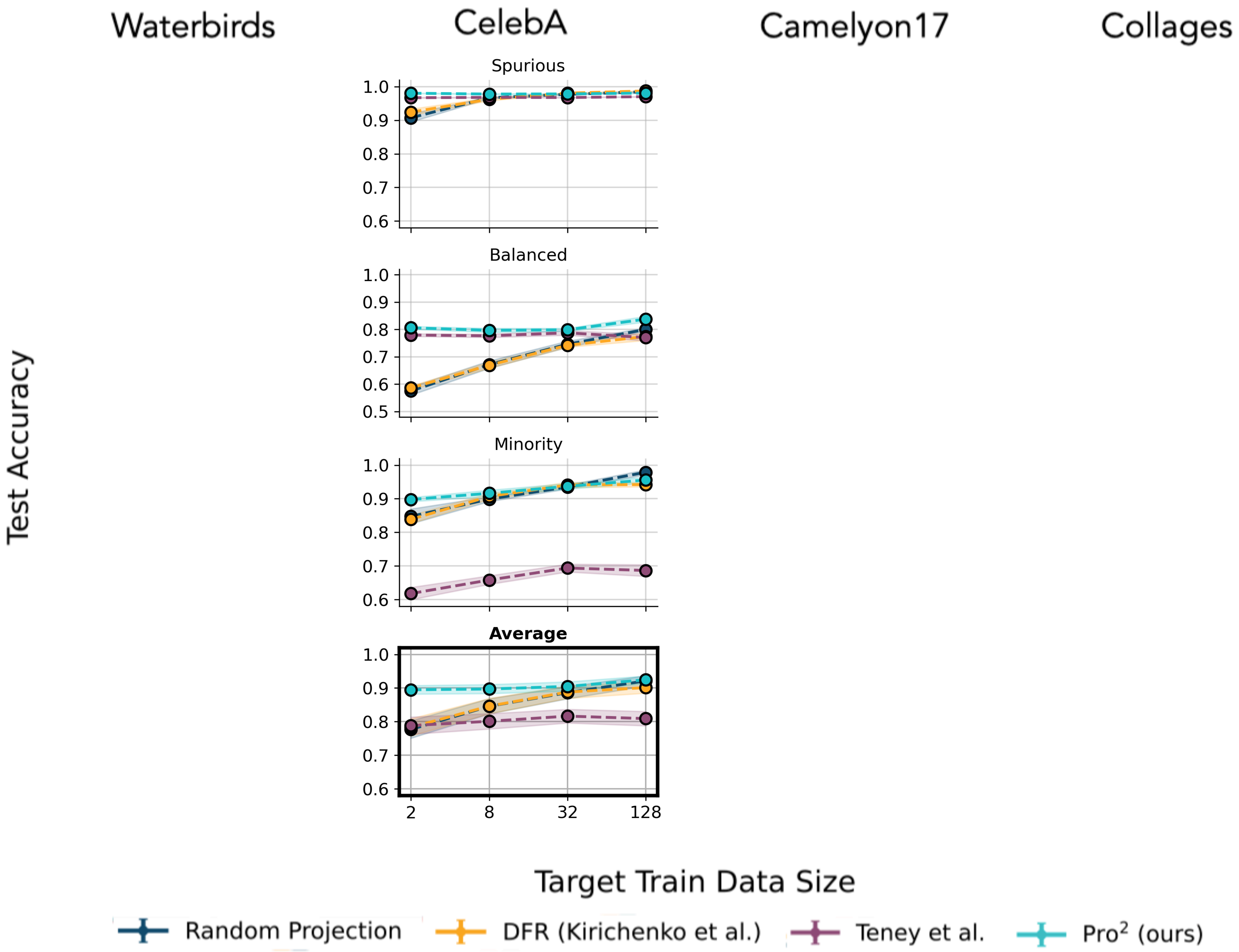
Pro² results in higher adaptation acc



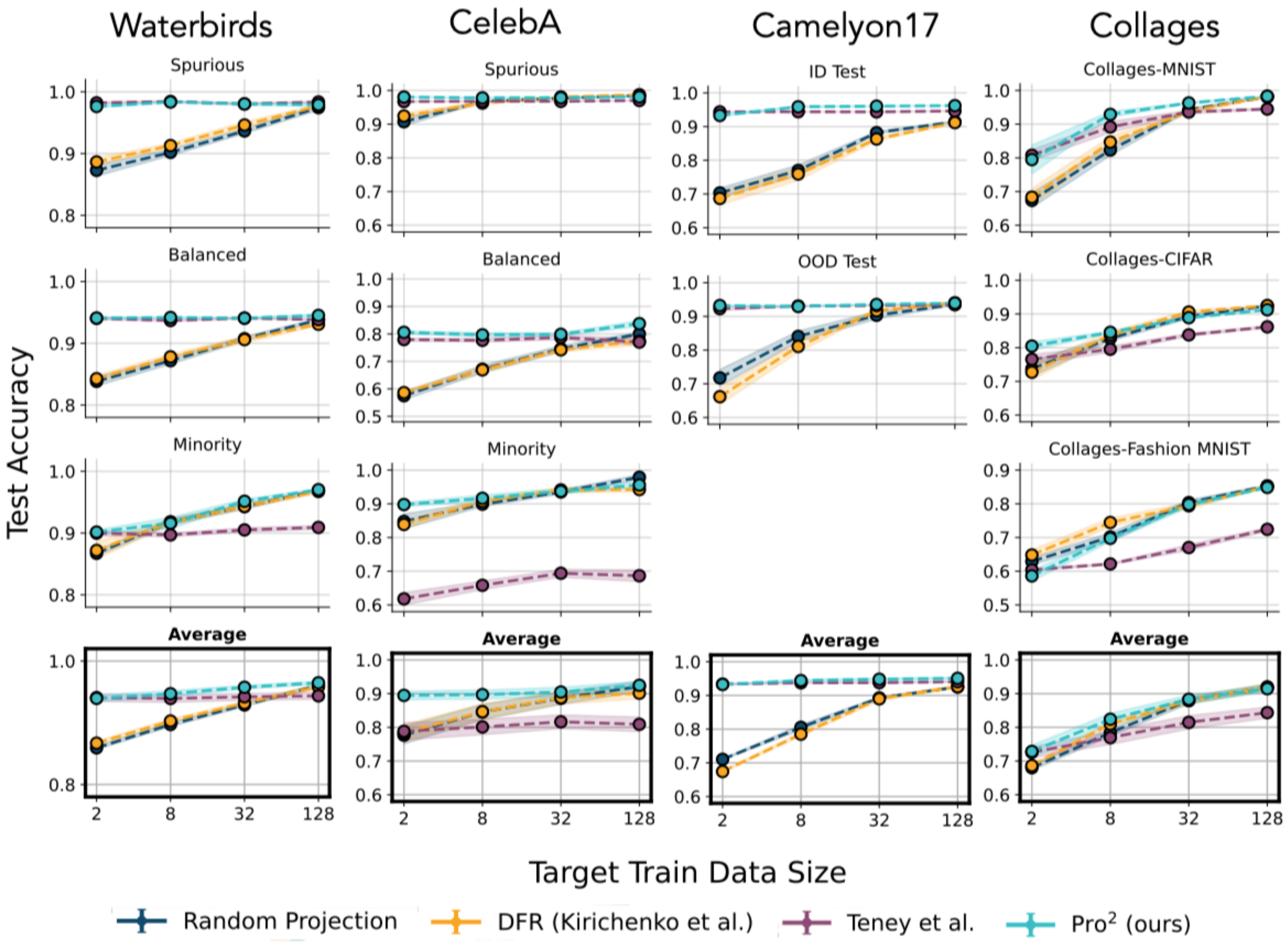
Pro² results in higher adaptation acc



Pro² results in higher adaptation acc



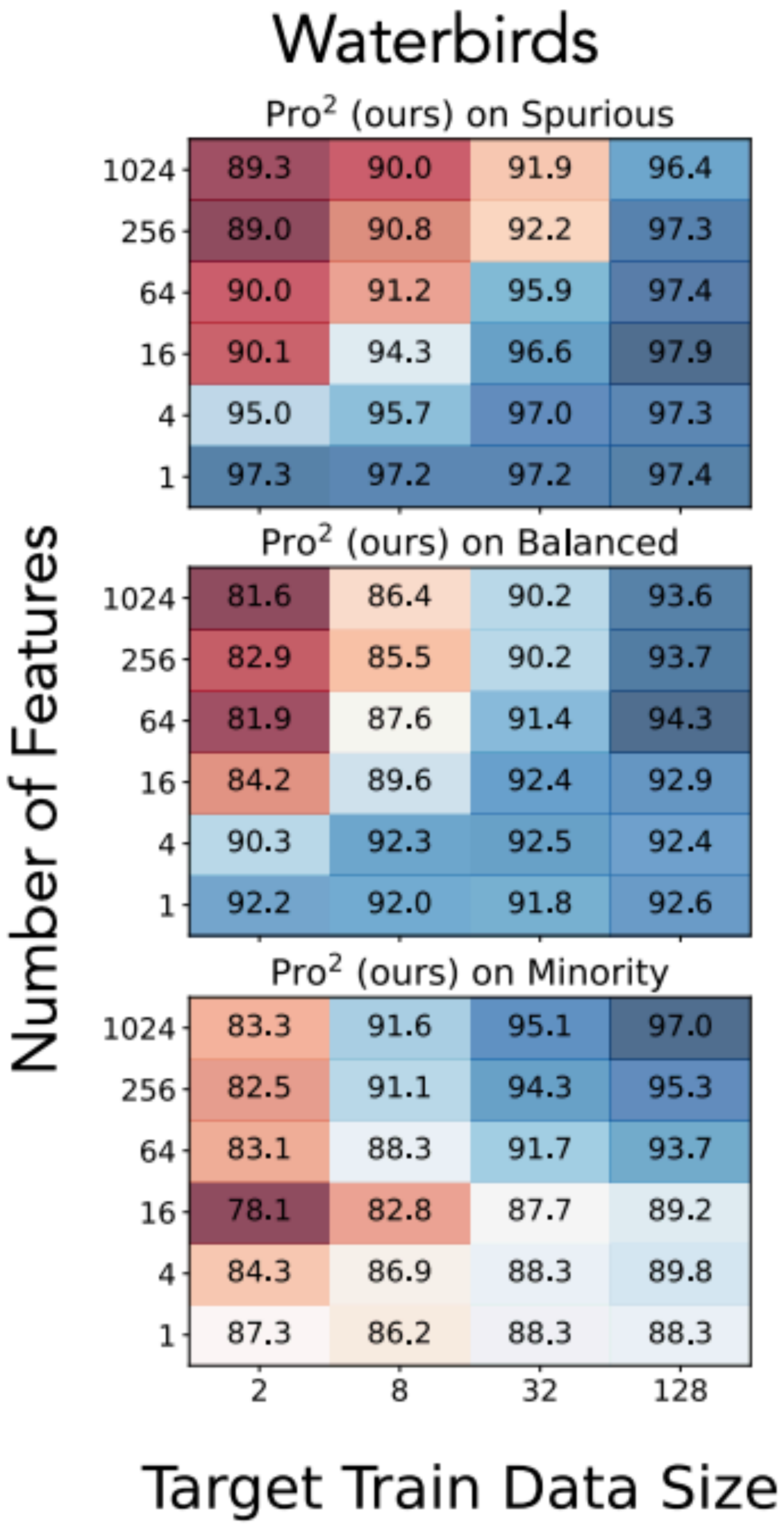
Pro² results in higher adaptation acc



Pro² bias-variance tradeoff

Smaller shifts

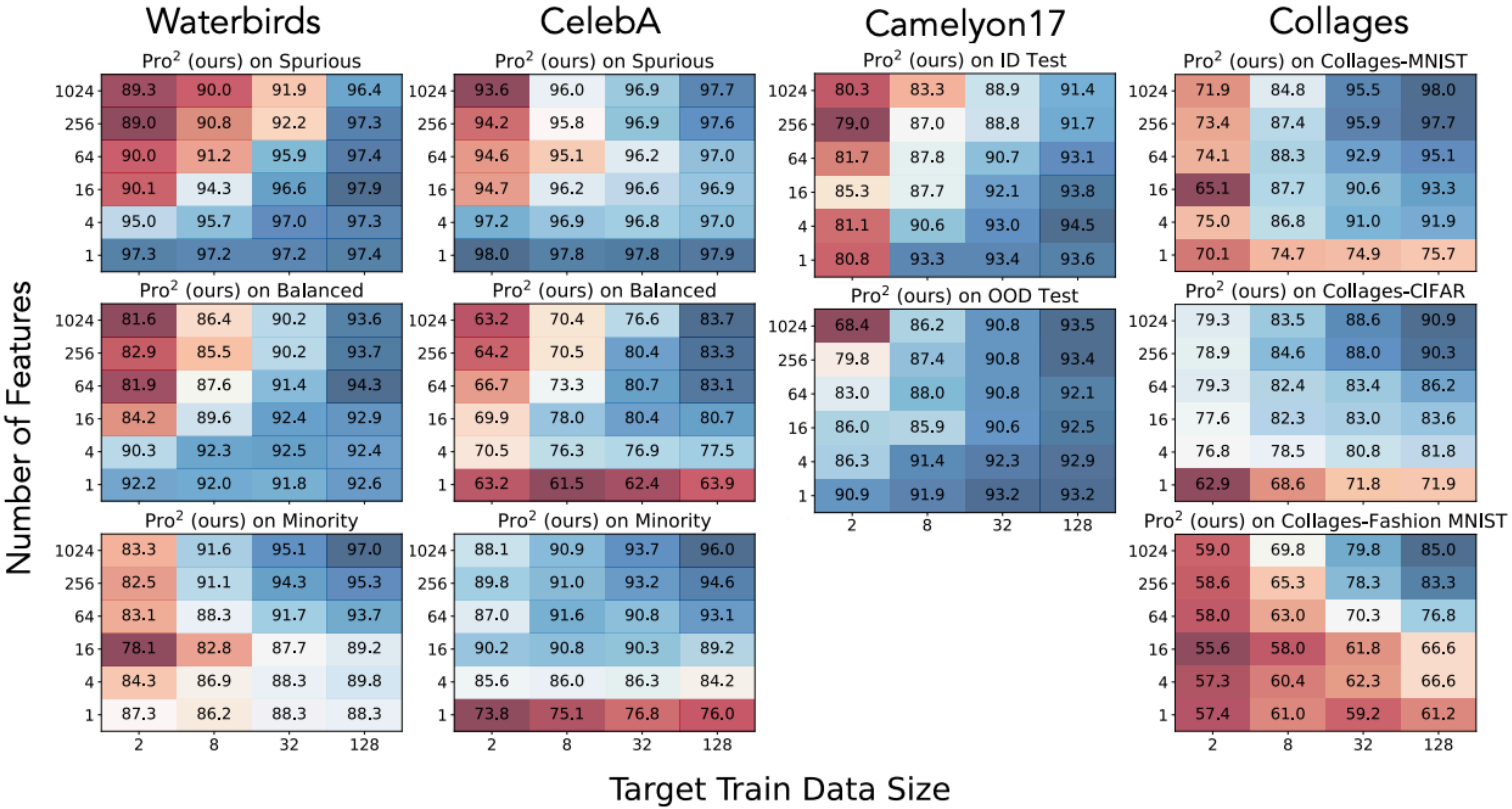
More severe shifts



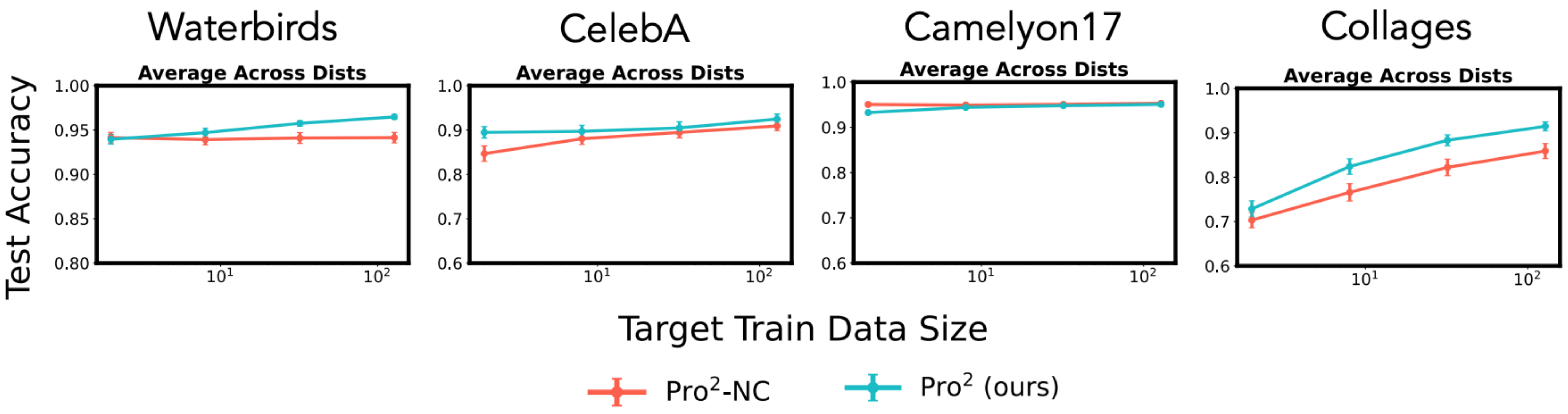
Pro² bias-variance tradeoff

Smaller shifts

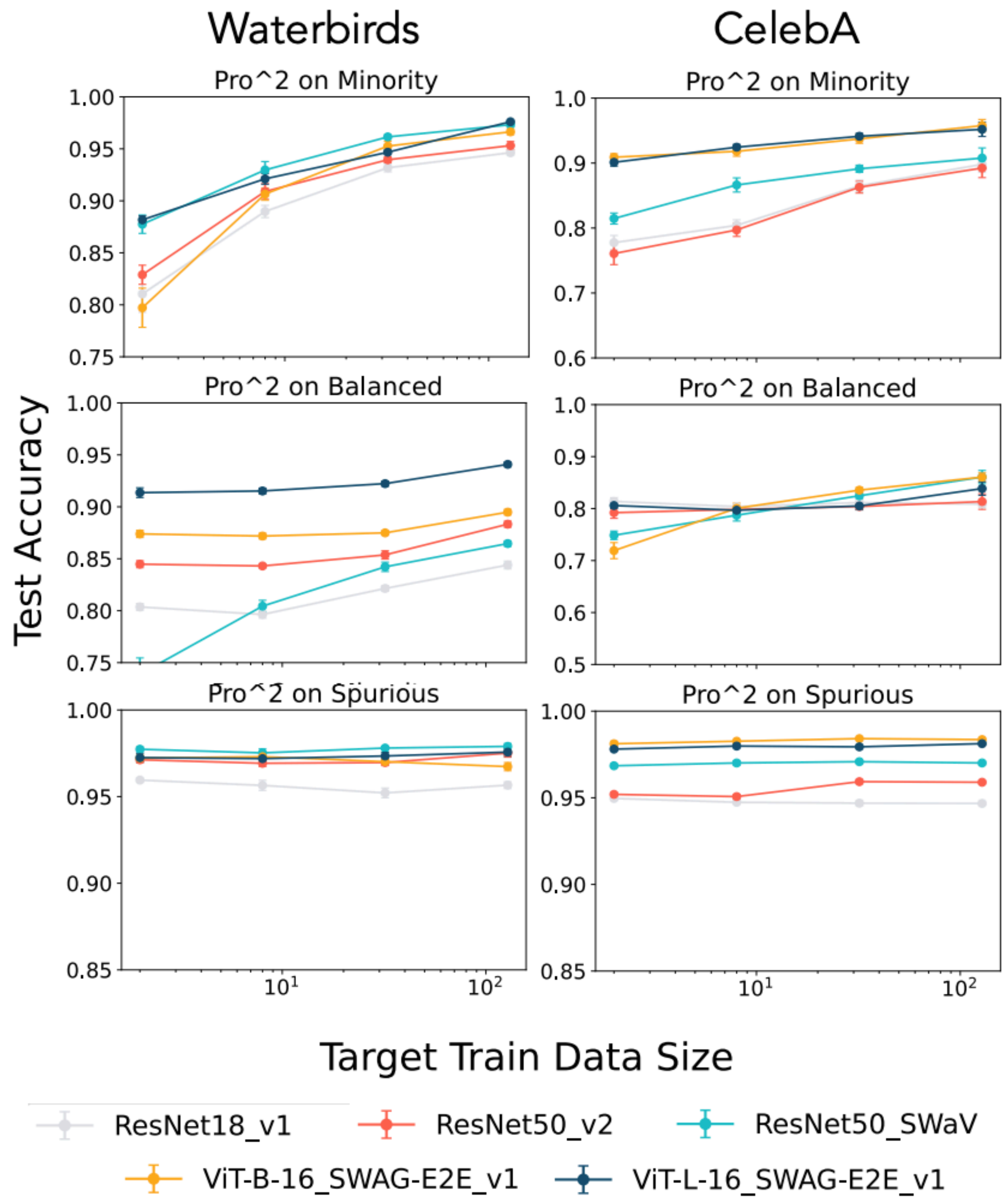
More severe shifts



Ablations



Orthogonality is important for learning a diverse set of features



Pro² improves with better pre-trained feature extractors

Takeaways

Pro2 is a lightweight, sample-efficient framework for adaptation.

Project: extract a diverse + predictive feature-space basis

Probe: interpolate to adapt to varying target distributions

Key Insight: The tradeoff between expressivity and inductive bias is critical in low data settings.

- Standard linear probing may not be best for few-shot adaptation.
- Pro2 better balances this tradeoff by learning diverse predictive features.

Future Work

Interesting future directions, including:

(1) Extending to *other problem settings*, such as active learning

(2) Exploring other methods to determine a *good feature-basis for adaptation*

(3) *Integrating with other fine-tuning methods* to further improve performance

(4) Select features to use in an *unsupervised* fashion

Thank you!

Paper: https://arxiv.org/pdf/2302.05441.pdf

Emails: asc8@stanford.edu and yoonho@stanford.edu

Annie S. Chen*

Yoonho Lee*

Amrith Setlur

Sergey Levine

Chelsea Finn