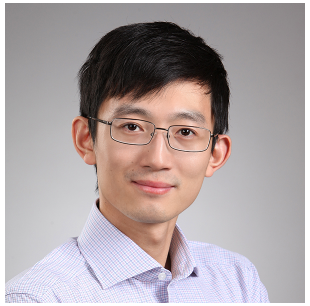
Fine-Tuning can Distort Pretrained Features and Underperform Out-of-Distribution

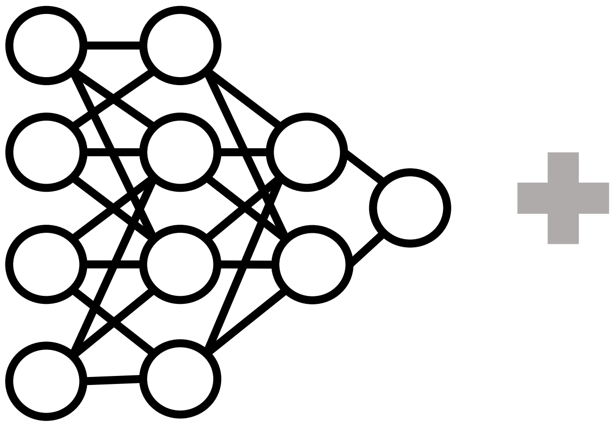



Ananya Kumar Aditi Raghunathan Robbie Jones Tengyu Ma Percy Liang

Classical ML: Train Model on Dataset

Satellite remote sensing task

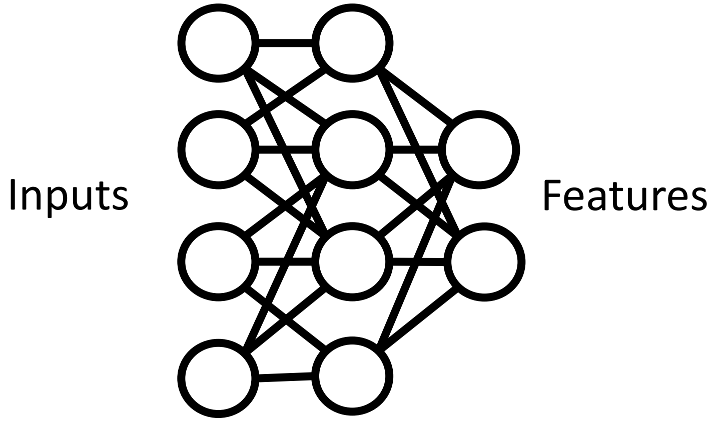


“In-distribution” training data

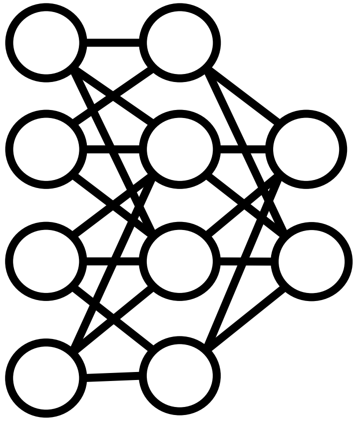
North America

Modern ML: *Adapt* Model on Dataset

Pretrained features



Satellite remote sensing task



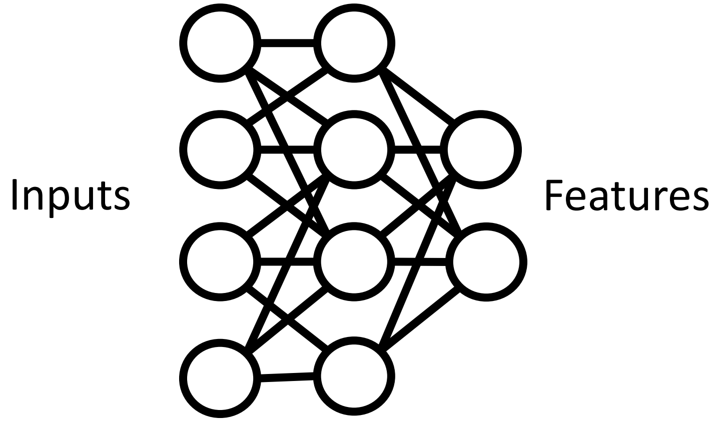
“In-distribution” training data

North America

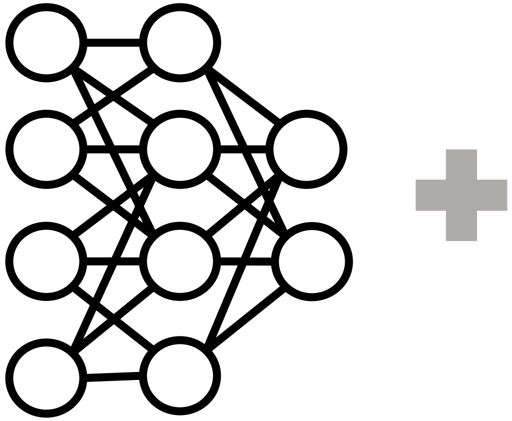
We start from pretrained models such as BERT (Devlin et al 2018), SimCLR (Chen et al 2020), CLIP (Radford et al 2021), and *adapt* them to our task---much better than training from scratch

Setting: Pretrain-Adapt-Test

Pretrained features



Satellite remote sensing



"In-distribution" training data

North America

"In-distribution" (ID) test data

North America

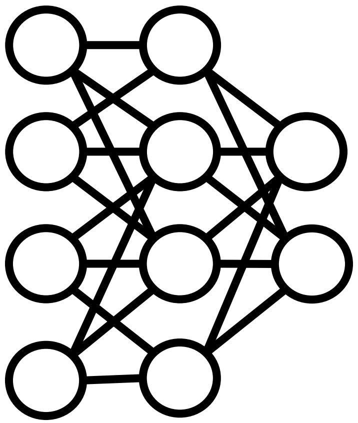
"Out-of-distribution" (OOD) test data

Africa

Setting: Pretrain-Adapt-Test

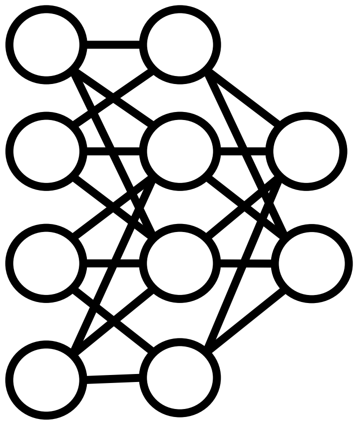
Pretrained features

Inputs



Features

Satellite remote sensing



+

"In-distribution" training data

North America

"In-distribution" (ID) test data

North America

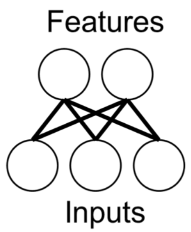
"Out-of-distribution" (OOD) test data

Africa

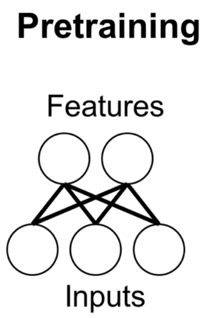
How should we adapt pretrained models (e.g. CLIP, SimCLR)?

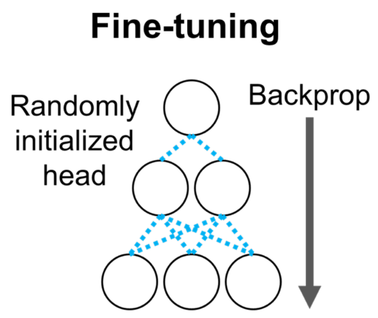
Linear Probing vs. Fine-tuning

Pretraining

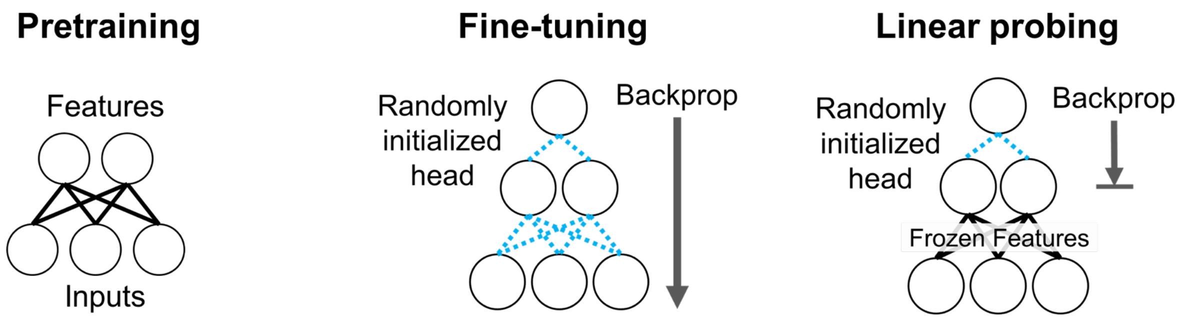


Linear Probing vs. Fine-tuning

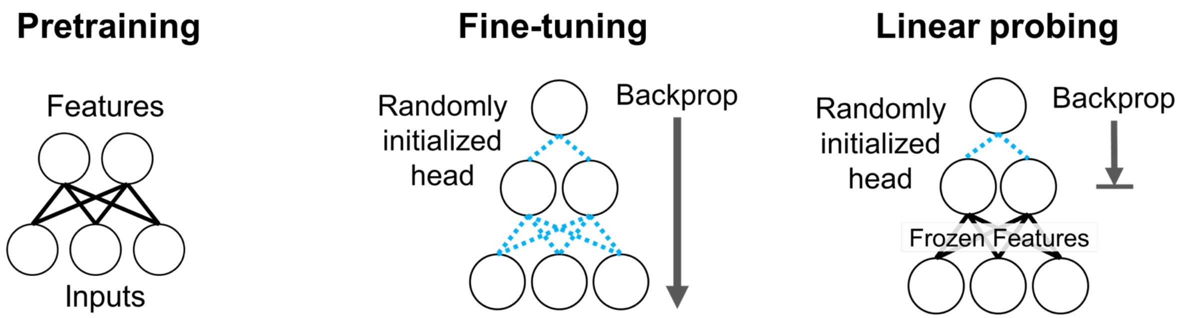




Linear Probing vs. Fine-tuning



Linear Probing vs. Fine-tuning



Which method does better?

Pop Quiz: Background, Living-17

	Cat	**Ape**	**Bear**
ID			
OOD			

Pop Quiz: Background, Living-17

- Breeds Living-17: task is to classify image into animal such as bear (ID contains black bears, sloth bears; OOD has brown bears, polar bears)
- Pretrained model: MoCo-V2 ResNet-50, seen *unlabeled* ImageNet images (including various types of bears)
- 17 classes of animals, around 50K training examples

Pop Quiz: Living-17

Living-17	**ID**	**OOD**
Scratch	92.4%	58.2%
Linear Probing	96.5%	?
Fine-Tuning	97.1%	

(a) LP < Scratch (b) Scratch < LP

Pop Quiz: Living-17

Living-17	**ID**	**OOD**
Scratch	92.4%	58.2%
Linear Probing	96.5%	82.2%
Fine-Tuning	97.1%	

(a) LP < Scratch **(b) Scratch < LP**

Pop Quiz: Living-17

Living-17	**ID**	**OOD**
Scratch	92.4%	58.2%
Linear Probing	96.5%	82.2%
Fine-Tuning	97.1%	?

(a) FT < Scratch (b) Scratch < FT < LP (c) LP < FT

Pop Quiz: Living-17

Living-17	**ID**	**OOD**
Scratch	92.4%	58.2%
Linear Probing	96.5%	82.2%
Fine-Tuning	97.1%	77.7%

(a) FT < Scratch **(b) Scratch < FT < LP** (c) LP < FT

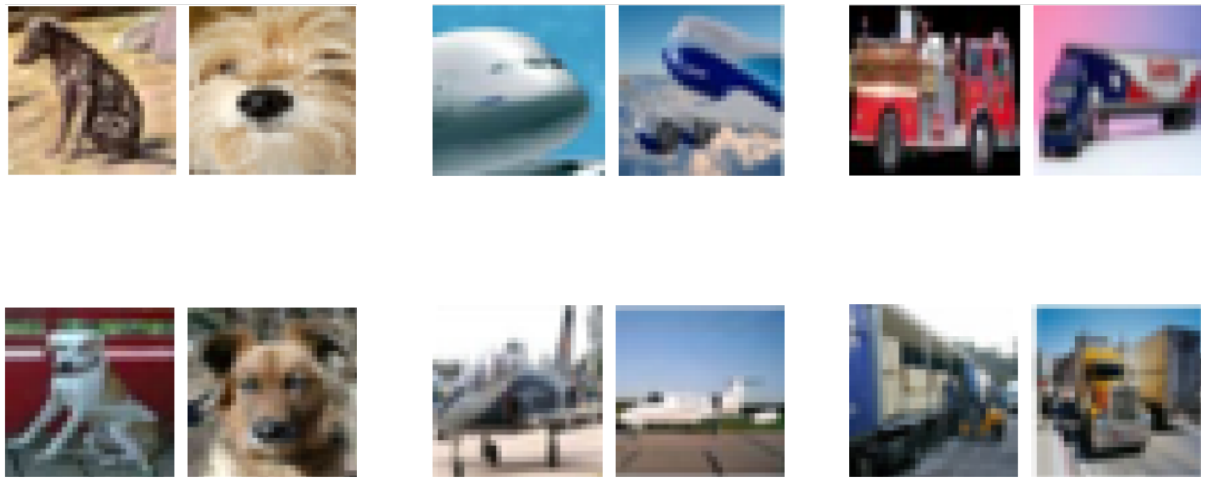
Pop Quiz: Background, CIFAR-10.1

- ID = CIFAR-10, OOD = CIFAR-10.1: Dataset collected using a similar protocol to CIFAR-10, “a minute distributional shift”

Dog **Plane** **Truck**

ID

OOD



Pop Quiz: CIFAR-10.1

CIFAR-10.1	**ID**	**OOD**
Linear Probing	91.8%	82.7%
Fine-Tuning	97.3%	?

(a) LP < FT (b) FT < LP

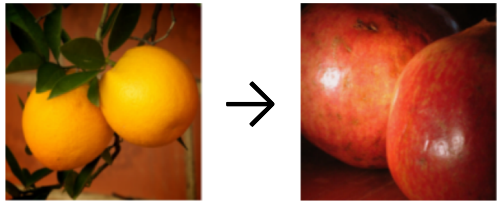
Pop Quiz: CIFAR-10.1

CIFAR-10.1	**ID**	**OOD**
Linear Probing	91.8%	82.7%
Fine-Tuning	97.3%	92.3%

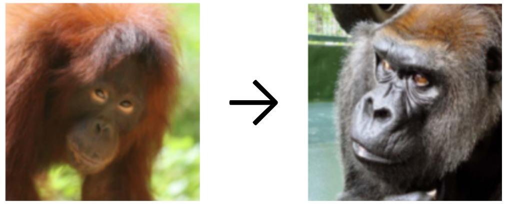
(a) LP < FT (b) FT < LP

Datasets

BREEDS-Entity-30



BREEDS-Living-17



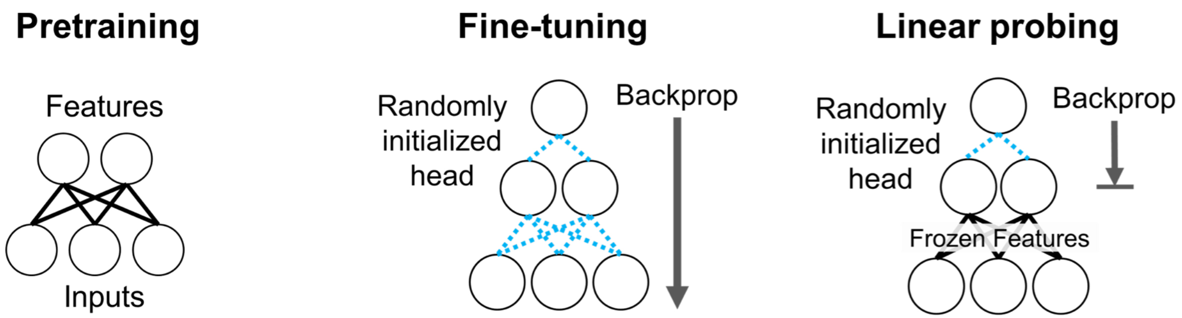
CIFAR-10 STL CIFAR-10.1

FMoW-America FMoW-Africa FMoW-Europe

DomainNet Sketch Real Painting Clipart

ImageNet ImNetV2 ImNet-R ImNet-Sketch ImNet-A

Linear Probing vs. Fine-tuning



ID test

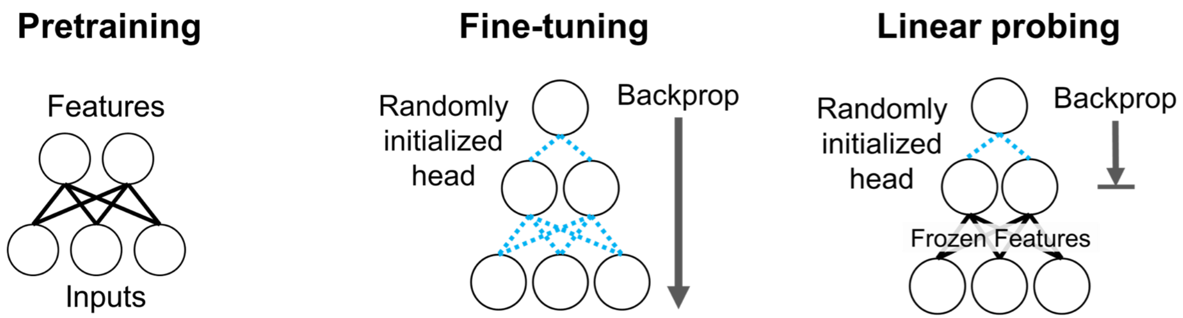
85.1%	82.9%

Average accuracies (7 ID datasets)

Common wisdom is fine-tuning works better than linear probing

(Kornblith et al 2019, Chen et al 2020, Zhai et al 2020, Chen et al 2021)

Linear Probing vs. Fine-tuning



ID test

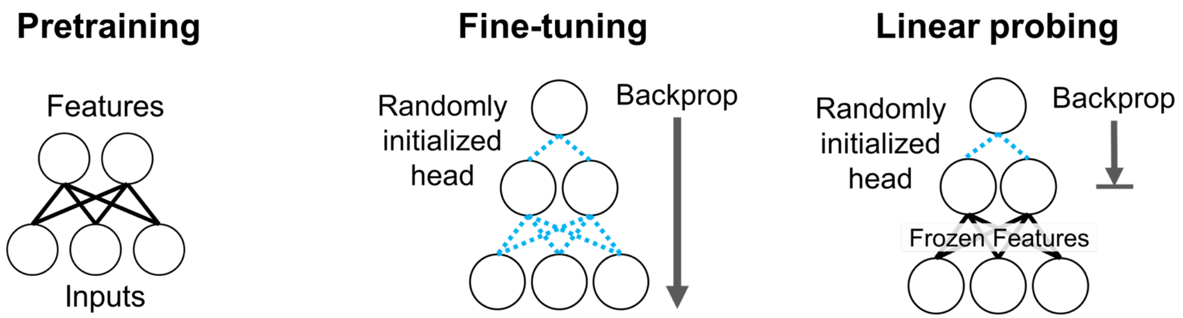
OOD test

	Fine-tuning	Linear probing
ID test	85.1%	82.9%
OOD test	59.3%	66.2%

Average accuracies (10 datasets)

Fine-tuning worse on 8/10 OOD datasets

Linear Probing vs. Fine-tuning



Fine-tuning can often do worse out-of-distribution

especially when the pretrained features are high quality and distribution shifts are large

Outline

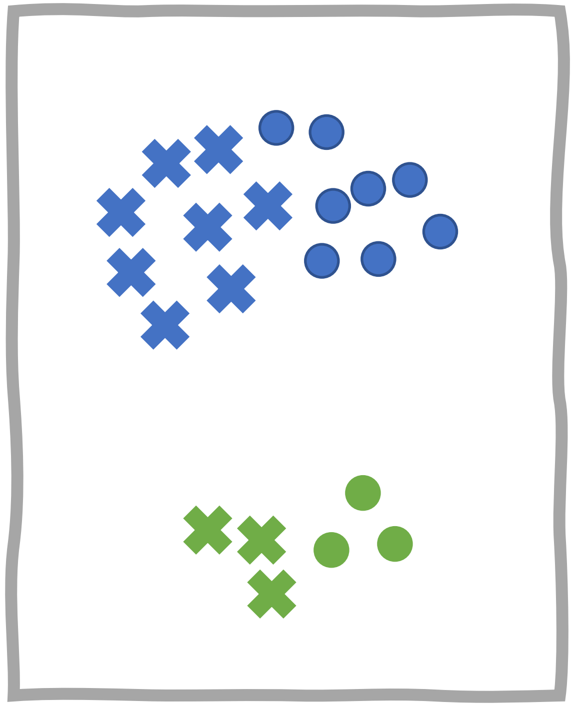
1. Fine-tuning can do worse than linear-probing OOD
2. Why fine-tuning can underperform OOD
3. Simple change to fine-tuning: improved accuracy on 10 datasets

Outline

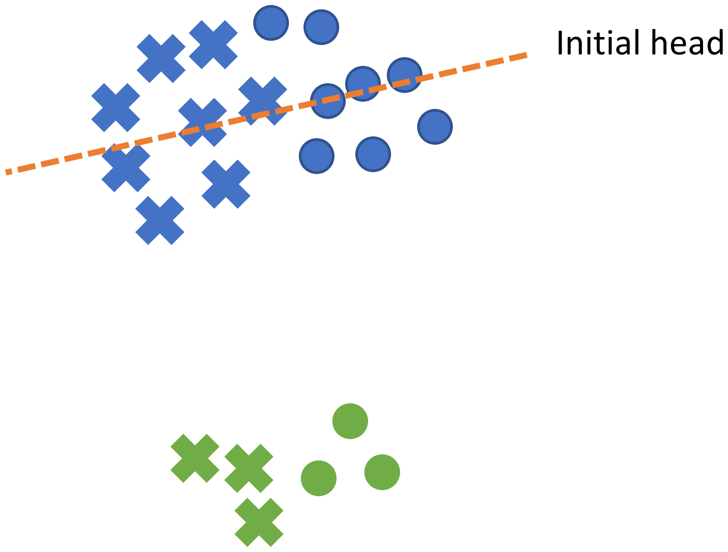
1. Fine-tuning can do worse than linear-probing OOD
2. **Why fine-tuning can underperform OOD**
3. Simple change to fine-tuning: improved accuracy on 10 datasets

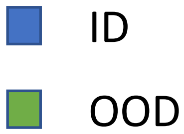
Feature Distortion Theory

Pretrained Features



Fine-tuning: features for ID examples change in sync with the linear head

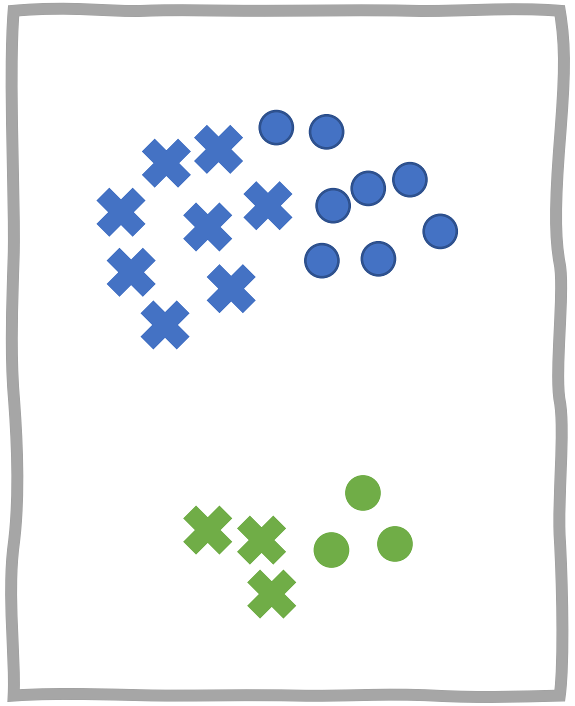




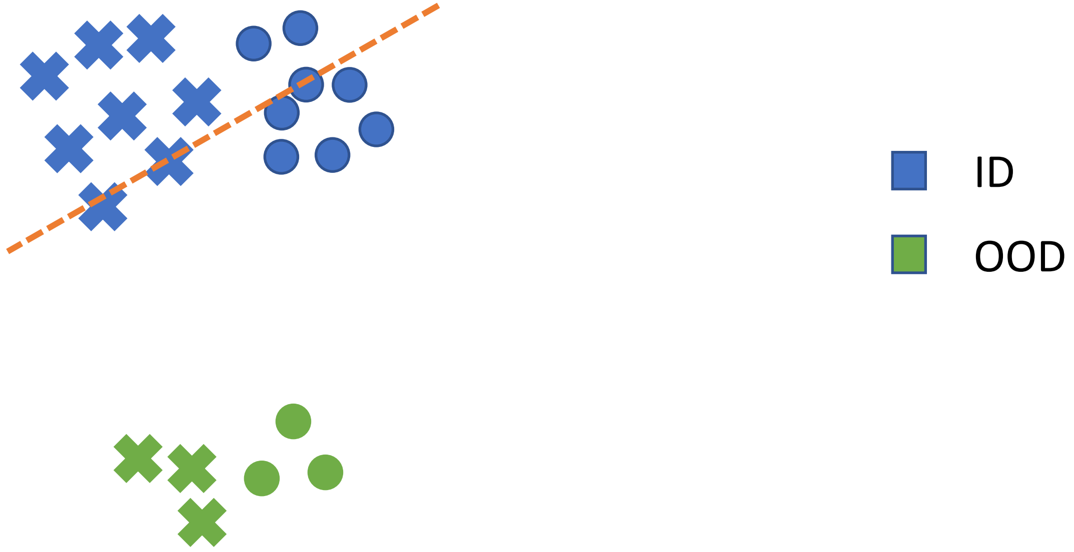
Features for OOD examples change less

Feature Distortion Theory

Pretrained
Features



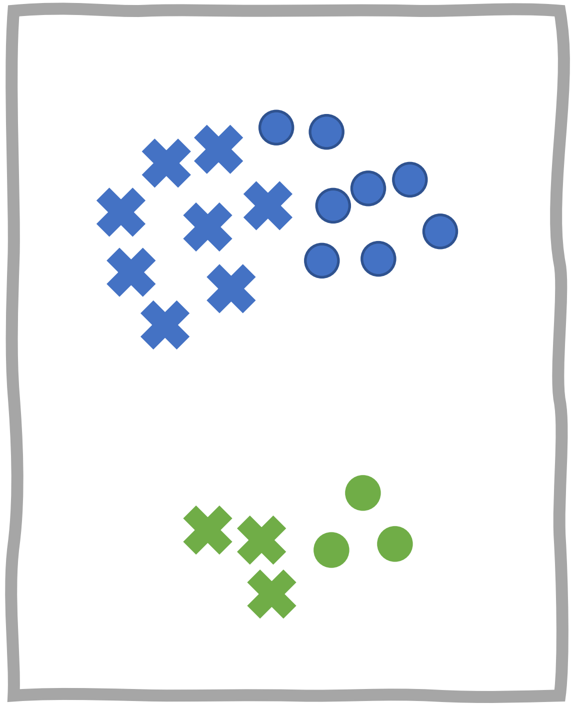
Fine-tuning: features for ID examples change
in sync with the linear head



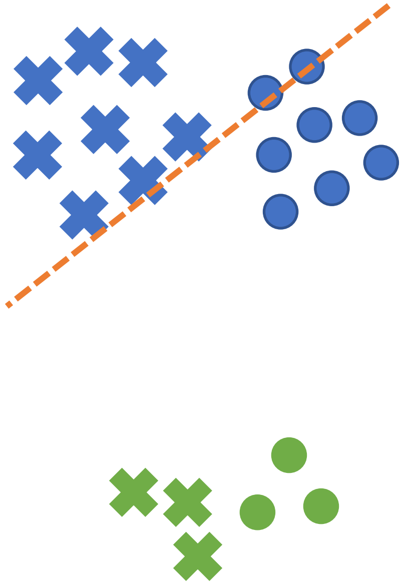
Features for OOD
examples change less

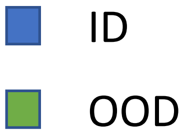
Feature Distortion Theory

Pretrained
Features



Fine-tuning: features for ID examples change
in sync with the linear head

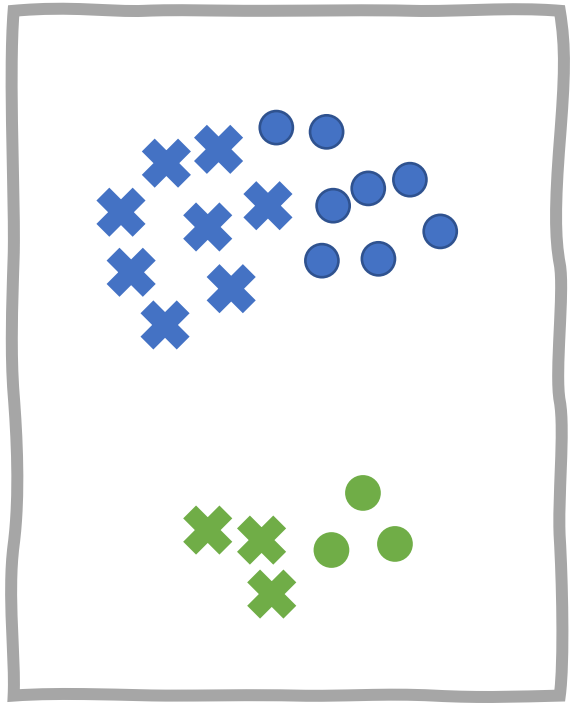




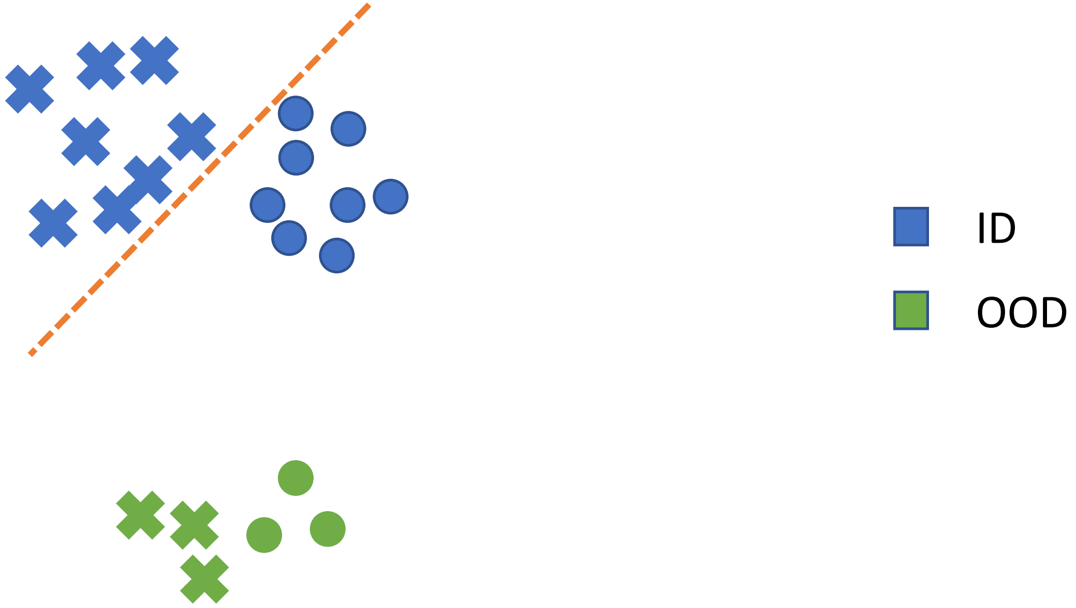
Features for OOD
examples change less

Feature Distortion Theory

Pretrained Features

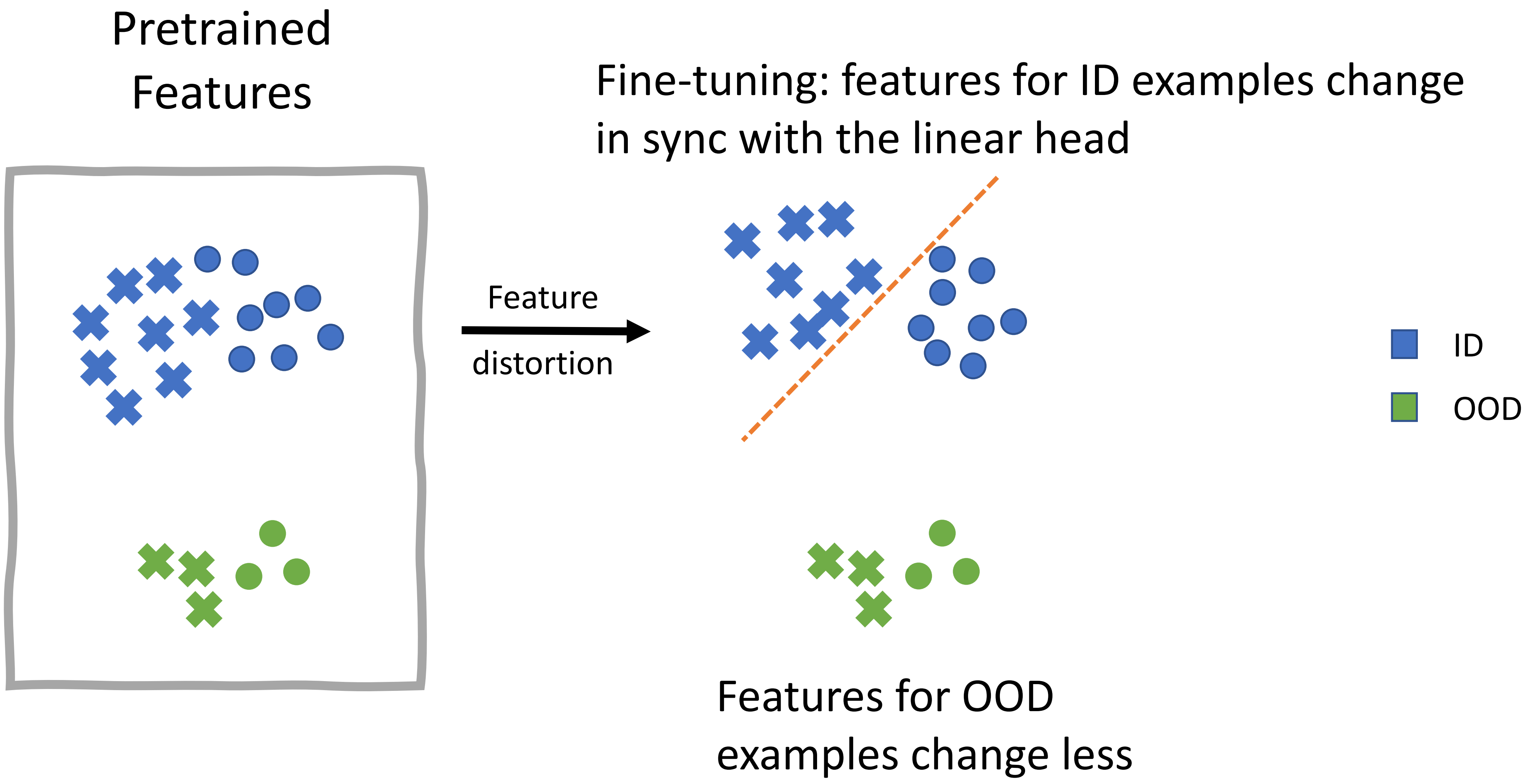


Fine-tuning: features for ID examples change in sync with the linear head



Features for OOD examples change less

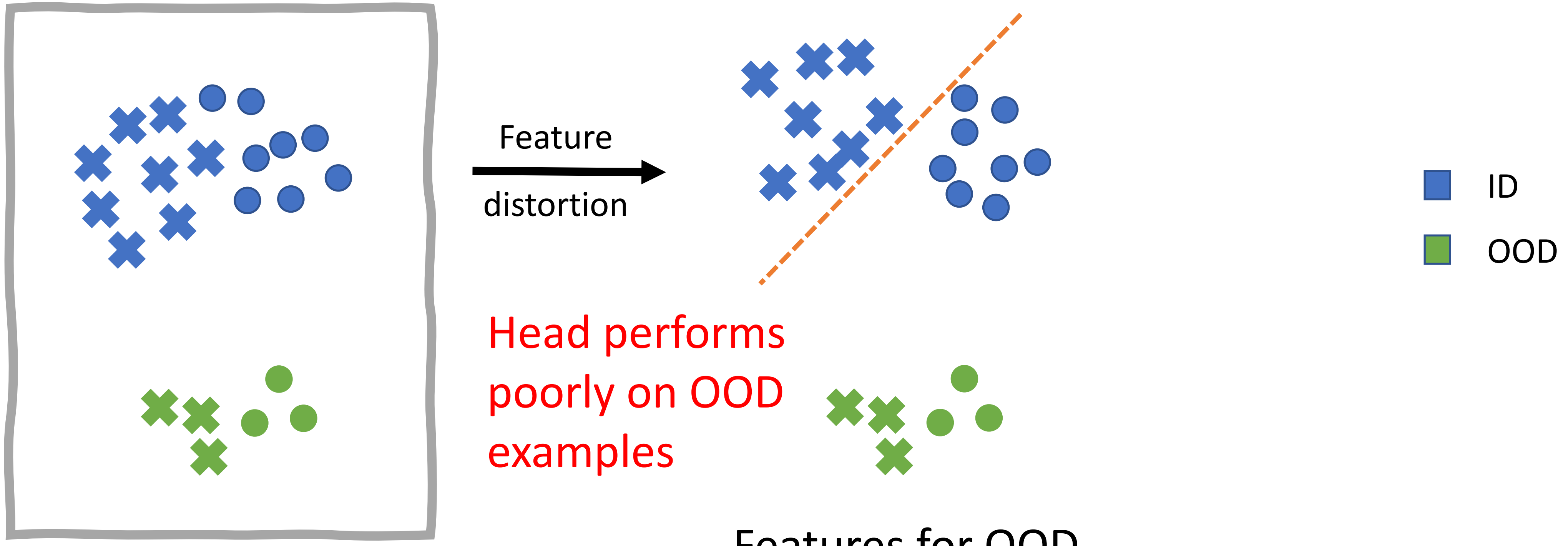
Feature Distortion Theory



Feature Distortion Theory

Pretrained
Features

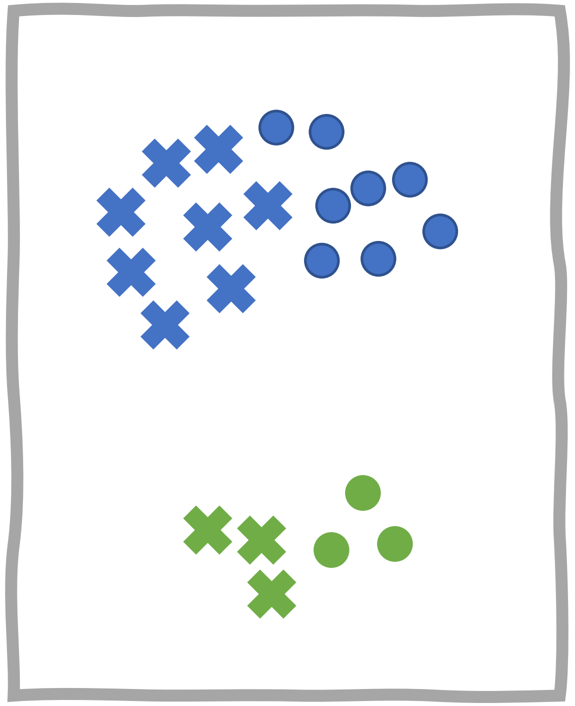
Fine-tuning: features for ID examples change
in sync with the linear head



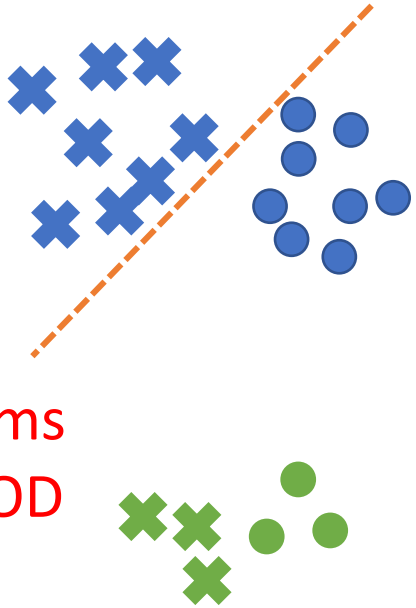
Features for OOD
examples change less

Feature Distortion Theory

Pretrained Features

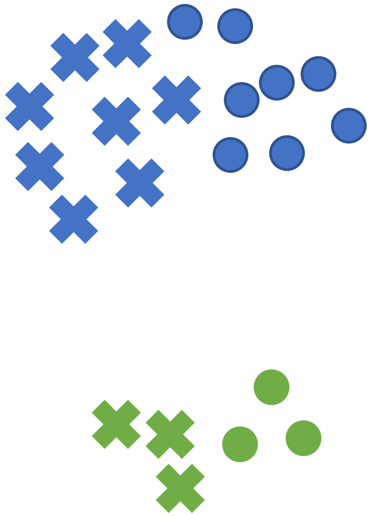


Fine-tuning



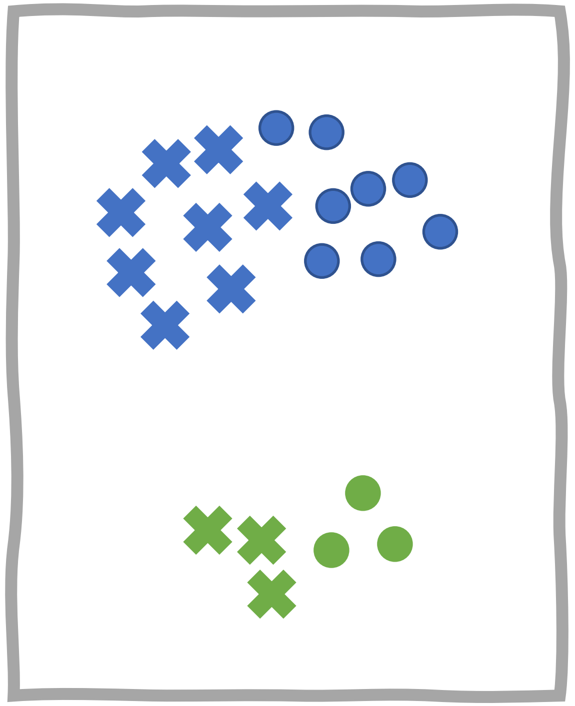
Head performs poorly on OOD examples

Linear probing: freezes pretrained features

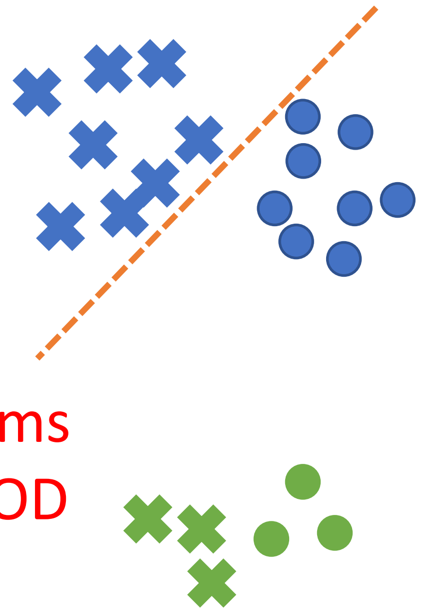


Feature Distortion Theory

Pretrained Features

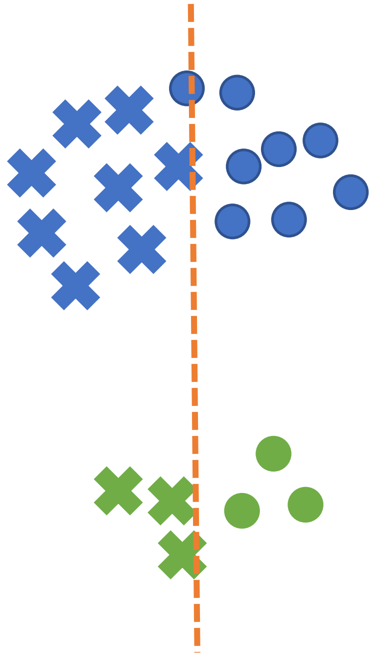


Fine-tuning



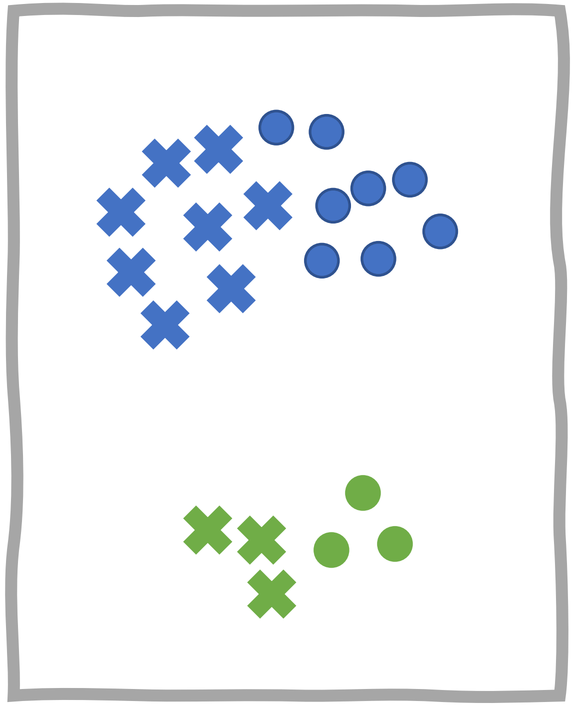
Head performs poorly on OOD examples

Linear probing: freezes pretrained features

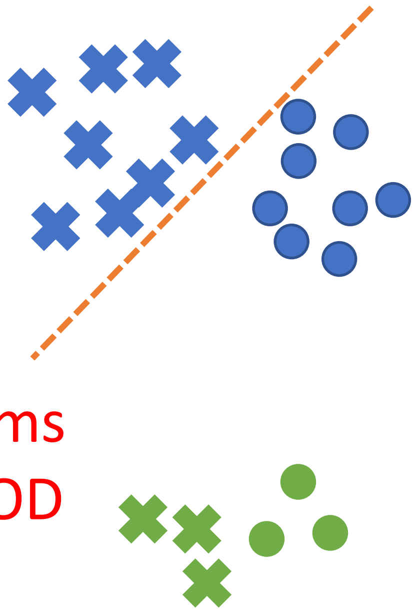


Feature Distortion Theory

Pretrained Features

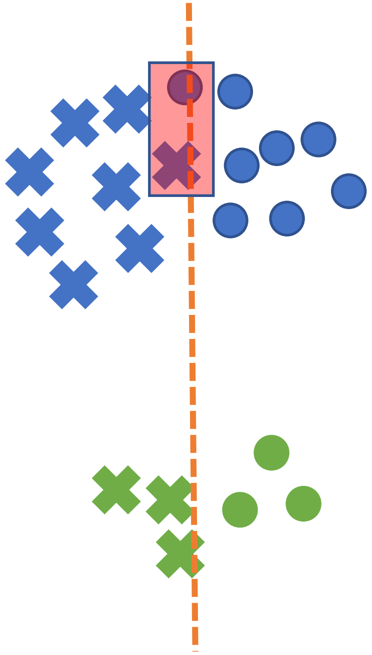


Fine-tuning



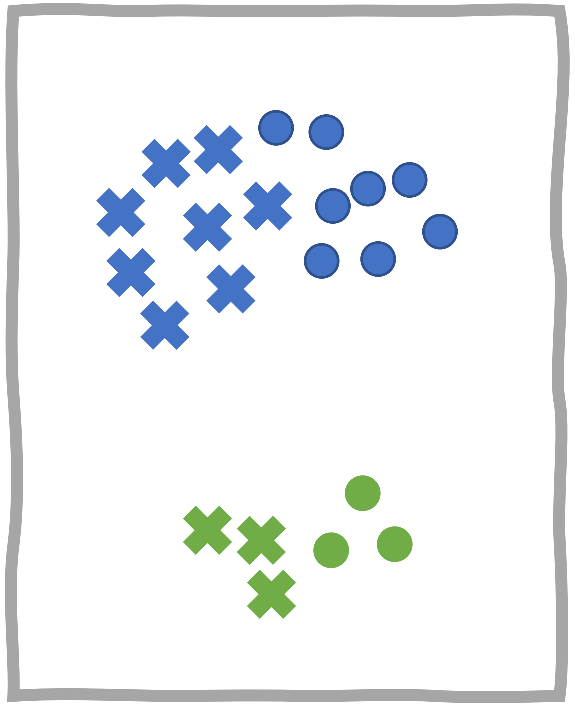
Head performs poorly on OOD examples

Linear probing: freezes pretrained features

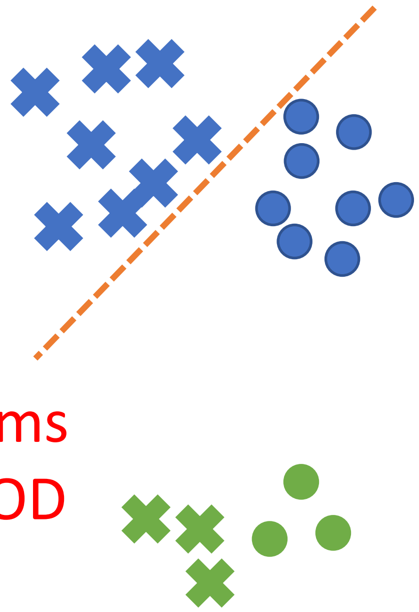


Feature Distortion Theory

Pretrained Features

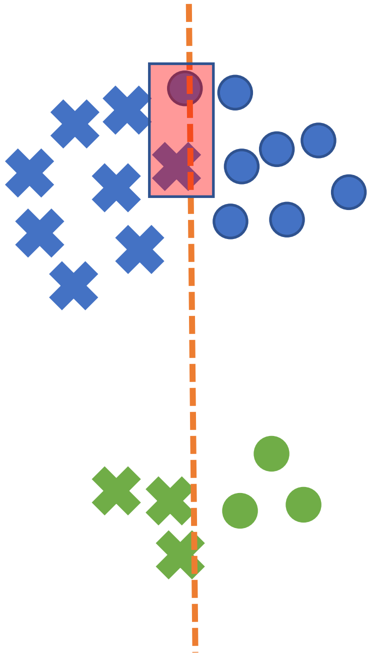


Fine-tuning



Head performs poorly on OOD examples

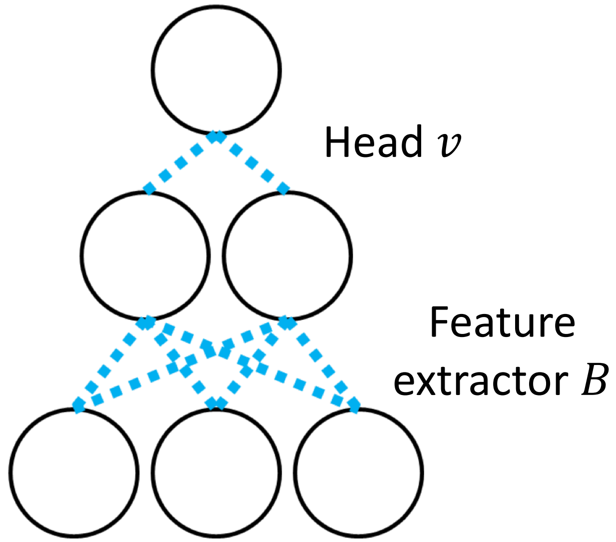
Linear probing: freezes pretrained features



Head is decent on OOD examples

Feature Distortion Theory

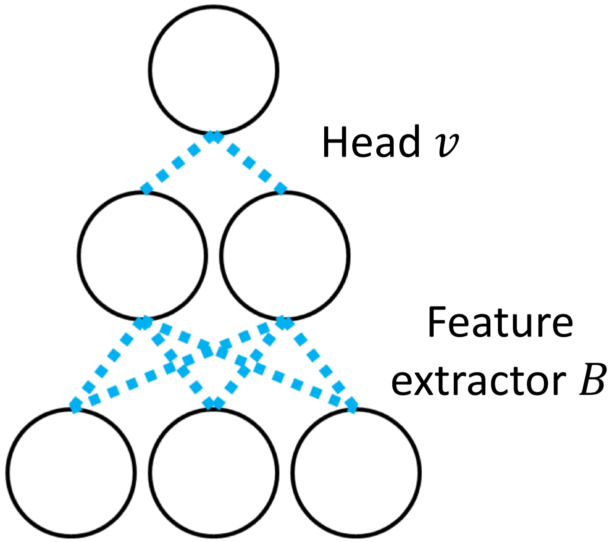
- Two-layer linear networks
  - High dimensional input: $x \in R^d$
  - Lower dimensional features: $B_* x \in R^k, k < d$
  - Ground truth outputs: $y = v_*^{\mathrm{T}} B_* x \in R$ (both ID and OOD)
- From prior work on pretraining, suppose we have B_0 close to B_*, so $\min_U ||B_* - UB_0||_2 \leq \epsilon$ where min is over rotation matrices U
- Let B_0, B_* have orthonormal rows
- $x_1, \ldots, x_n \in R^d$ are training examples with, $S = \mathrm{span}(\{x_1, \ldots, x_n\})$



Feature Distortion Theory

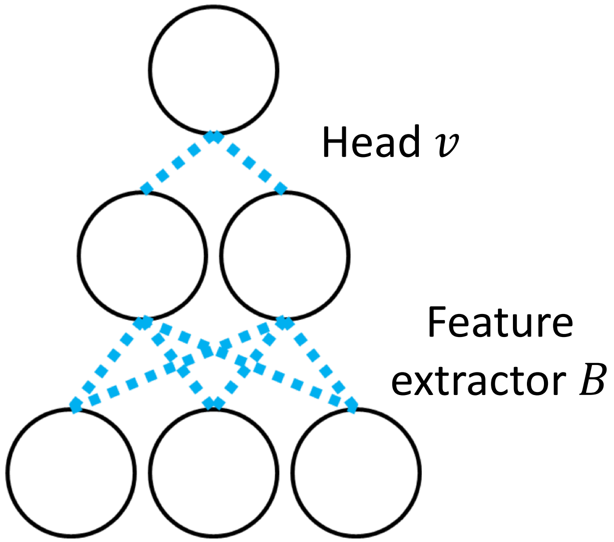
- $y = v_*^{\mathrm{T}} B_* x$ (both ID and OOD) where $x \in R^d, B_* x \in R^k$
- Have B_0 close to B_* (from pretraining)

- Squared loss: $\hat{L}(v, B) = \frac{1}{n}\sum_i (y_i - v^{\mathrm{T}} B x_i)^2$
- Fine-tuning: update v, B via gradient flow (non-convex)
  - $\partial_t v_{\mathrm{ft}}(t) = -\nabla_v \hat{L}(v, B)$ and $\partial_t B_{\mathrm{ft}}(t) = -\nabla_B \hat{L}(v, B)$
- Linear probing: update v via gradient flow (convex)
  - $\partial_t v_{\mathrm{lp}}(t) = -\nabla_v \hat{L}(v, B)$ and $\partial_t B_{\mathrm{lp}}(t) = 0$
- Initialize both with $B_{\mathrm{ft}}(0) = B_{\mathrm{lp}}(0) = B_0$ and $v_{\mathrm{ft}}(0) = v_{\mathrm{lp}}(0) = v_0$ where $v_0 = 0$ or $v_0 \sim N(0, \sigma^2 I_k)$



Feature Distortion Theory

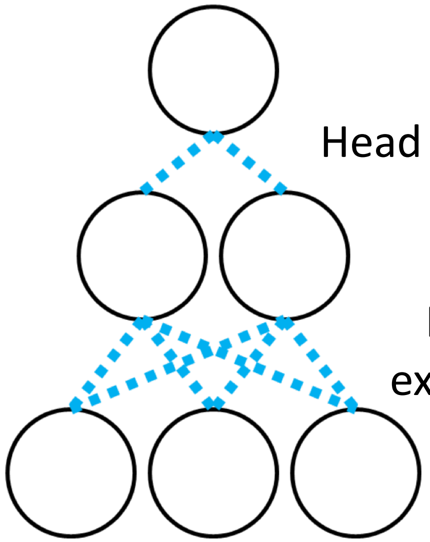
- $y = v_*^{\mathrm{T}} B_* x$ (both ID and OOD) where $x \in R^d, B_* x \in R^k$
- Have B_0 close to B_* (from pretraining)
- $v_{\mathrm{lp}}, v_{\mathrm{ft}}, B_{\mathrm{ft}}$ from gradient flow on training data



- OOD evaluation:
  - P_{ood} has invertible covariance matrix Σ
  - $L_{\mathrm{ood}} = E_{x \sim P_{\mathrm{ood}}}[(y - v^{\top} B x)^2]$
- Overparameterized: $1 \le \dim(S) \le d - k$
- Intuition: OOD includes directions not seen in training data. Both fine-tuning and training from scratch fit train loss, but have different test losses

Feature Distortion Theory

- $y = v_*^{\mathrm{T}} B_* x$ (both ID and OOD) where $x \in R^d, B_* x \in R^k$
- Have B_0 close to B_* (from pretraining)
- $v_{\mathrm{lp}}, v_{\mathrm{ft}}, B_{\mathrm{ft}}$ from gradient flow on training data
- L_{ood}, OOD loss, includes unseen directions



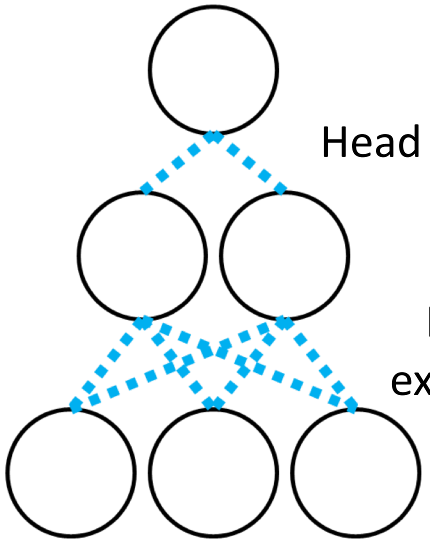
<u>Theorem 3.3 (FT error, simplified & informal)</u>

$$L_{\mathrm{ood}}(v_{\mathrm{ft}}(t), B_{\mathrm{ft}}(t)) \geq O\left(\frac{\alpha}{k}\varphi\right) \textit{ for small } \epsilon$$

- $\varphi^2 = |(v_0^{\mathrm{T}} v_*)^2 - (v_*^{\mathrm{T}} v_*)^2|$ is the initial head alignment error
- $\alpha = \cos\theta_{\max}(S^{\perp}, R_0)$ where $R_0 = \mathrm{rowspace}(B_0)$ which we assume is non-zero

Feature Distortion Theory

- $y = v_*^{\mathrm{T}} B_* x$ (both ID and OOD) where $x \in R^d, B_* x \in R^k$
- Have B_0 close to B_* (from pretraining)
- $v_{\mathrm{lp}}, v_{\mathrm{ft}}, B_{\mathrm{ft}}$ from gradient flow on training data
- L_{ood}, OOD loss, includes unseen directions



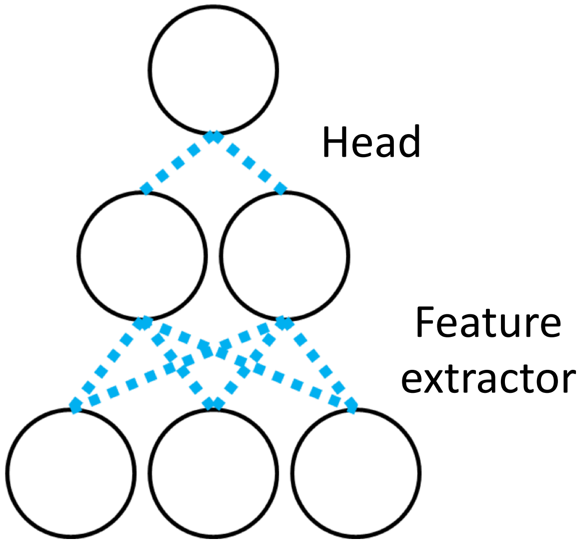
<u>Theorem 3.5 (LP vs. FT OOD, informal)</u>

$$\forall t, \qquad \frac{L_{\mathrm{ood}}\left(v_{\mathrm{lp}}^{\infty}, B_0\right)}{L_{\mathrm{ood}}\left(v_{\mathrm{ft}}(t)\ B_{\mathrm{ft}}(t)\right)} \xrightarrow{p} 0, \qquad \text{as } B_0 \to B_* \text{ (up to rotations)}$$

- Assume $\cos\theta_{\max}(S, R_*), \cos\theta_{\max}(S^{\perp}, R_*) \neq 0$ where $R_* = \mathrm{rowspace}(B_*)$

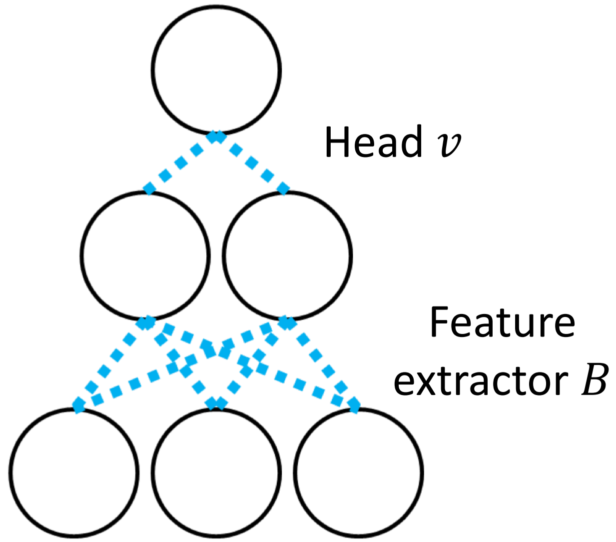
Feature Distortion Theory

- Suppose training data sampled from P_{id}, supported and with density on m-dimensional subspace S with $d - k > m > k$ and $n \geq m$
- OOD: fine-tuning worse than linear probing
  - If pretrained features good, OOD shift large
  - Throughout the process of fine-tuning
- ID: fine-tuning better than linear probing

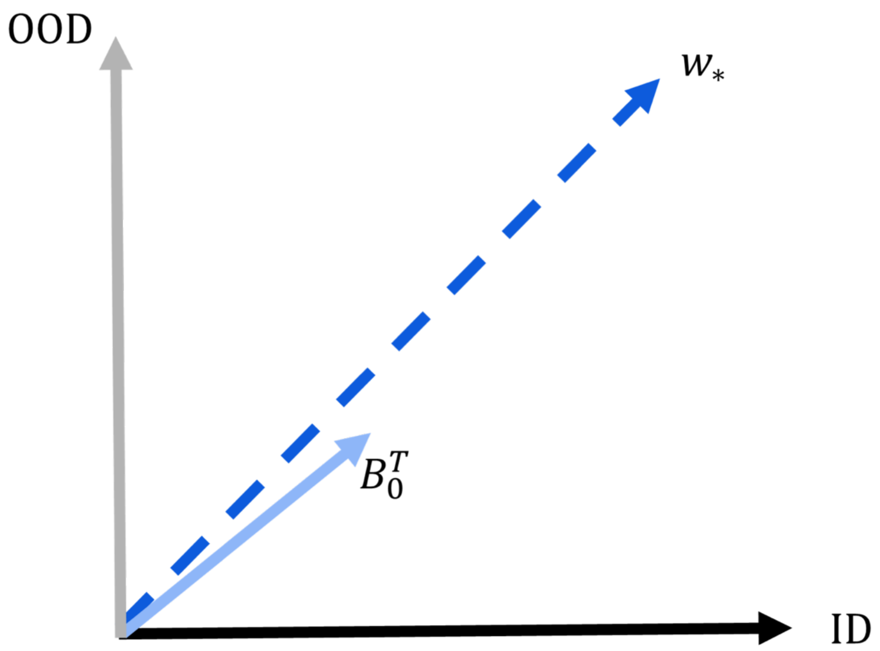


Feature Distortion Theory

- Prior work studies linear probing (fitting linear head on features)
- Fine-tuning is non-convex, trajectory is complicated and has no known closed form even for two-layer linear networks
- Tool: leverage invariants that hold throughout process of fine-tuning

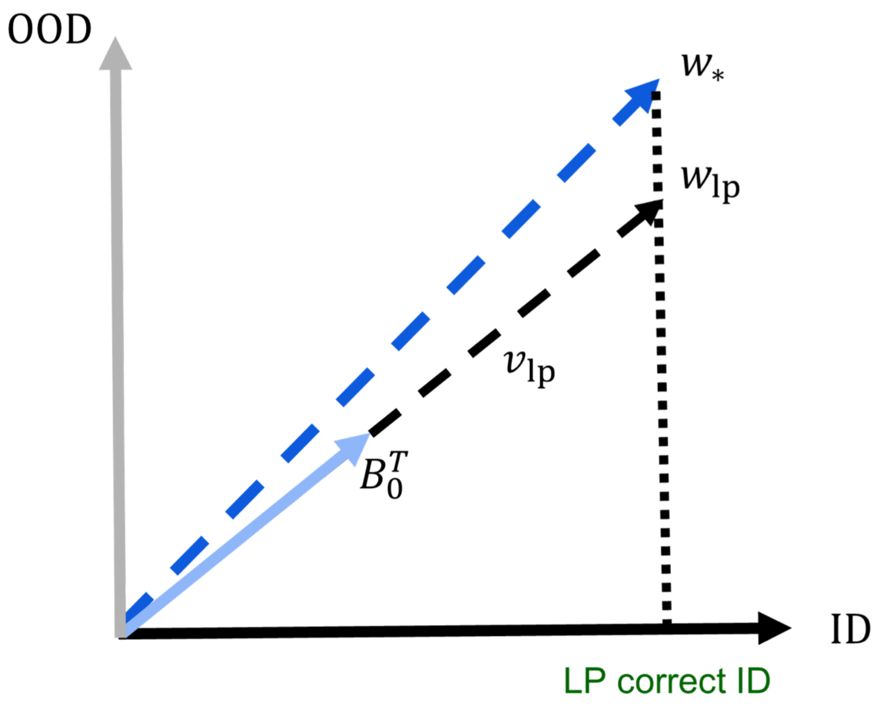


Feature Distortion (Toy Example)



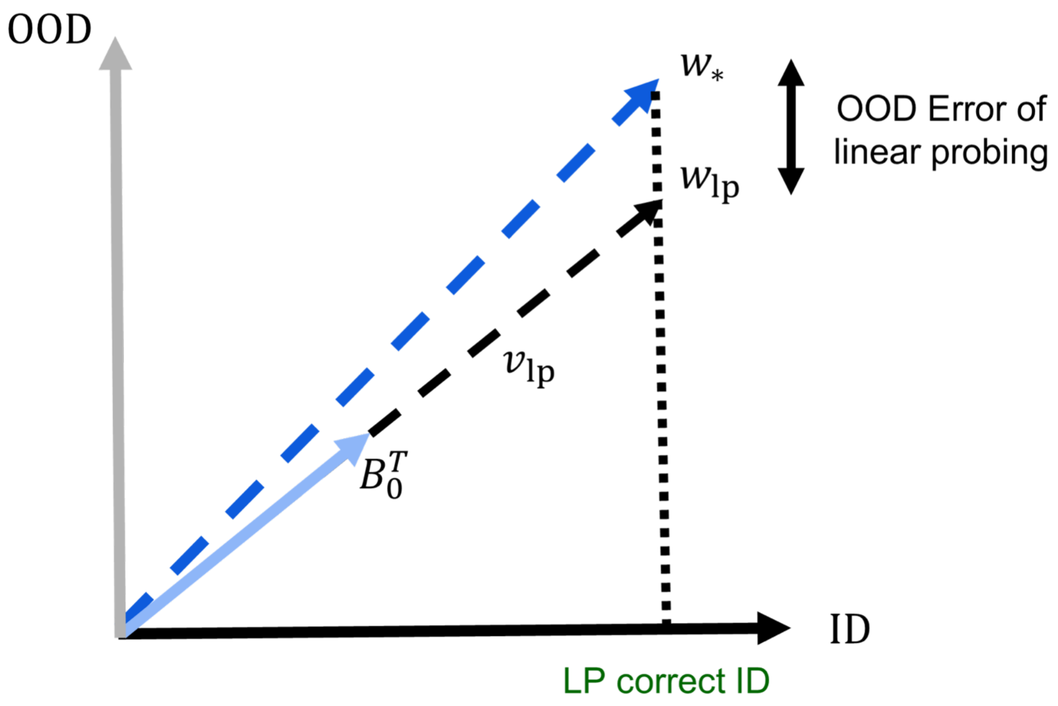
(a) Toy example (Linear probing)

Feature Distortion (Toy Example)



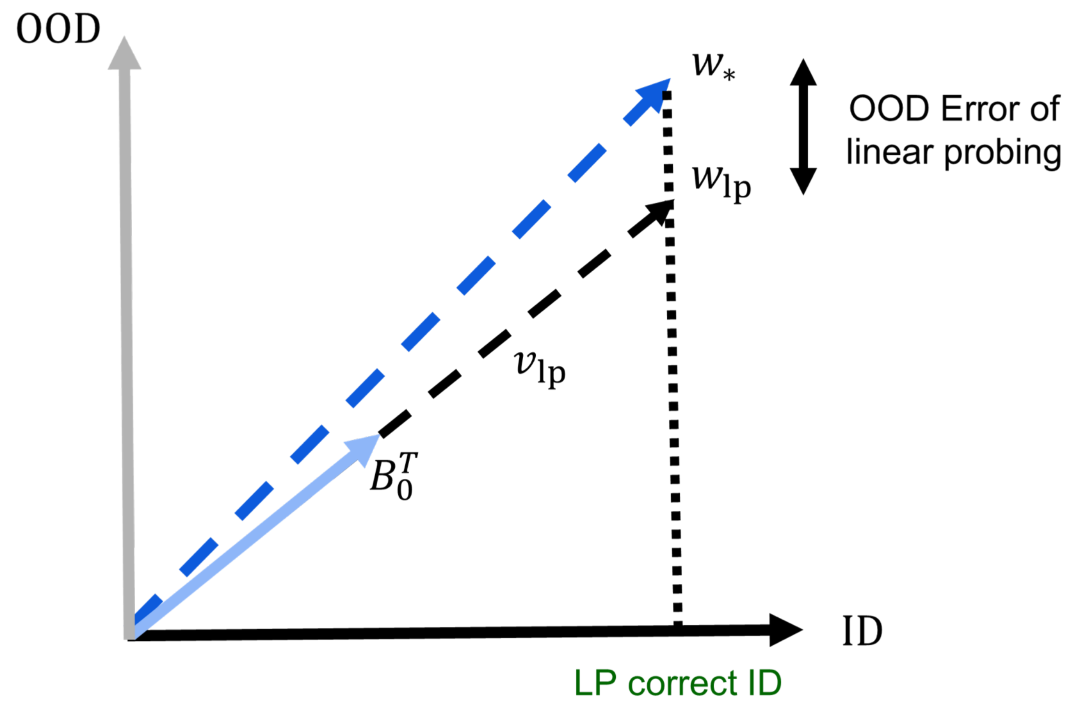
(a) Toy example (Linear probing)

Feature Distortion (Toy Example)

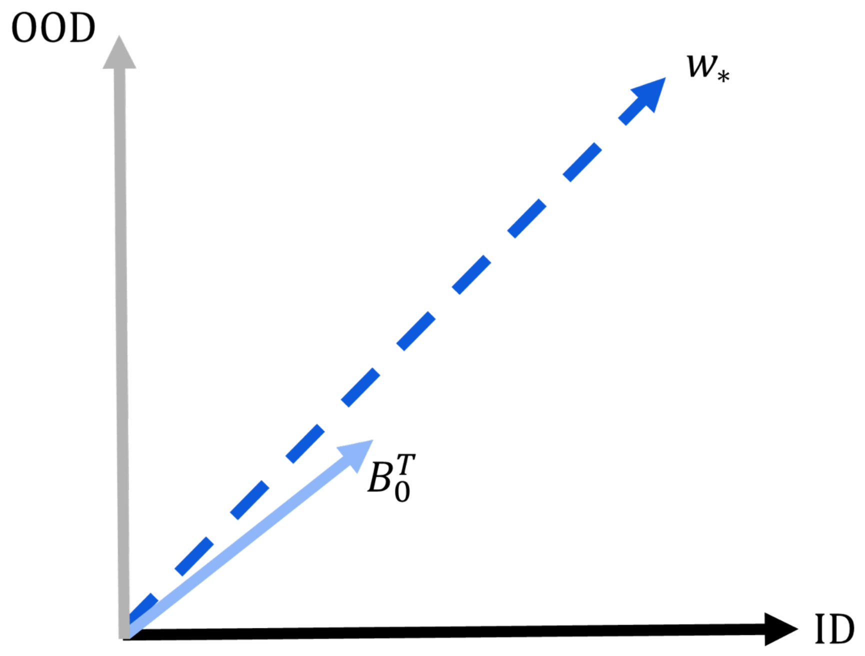


(a) Toy example (Linear probing)

Feature Distortion (Toy Example)

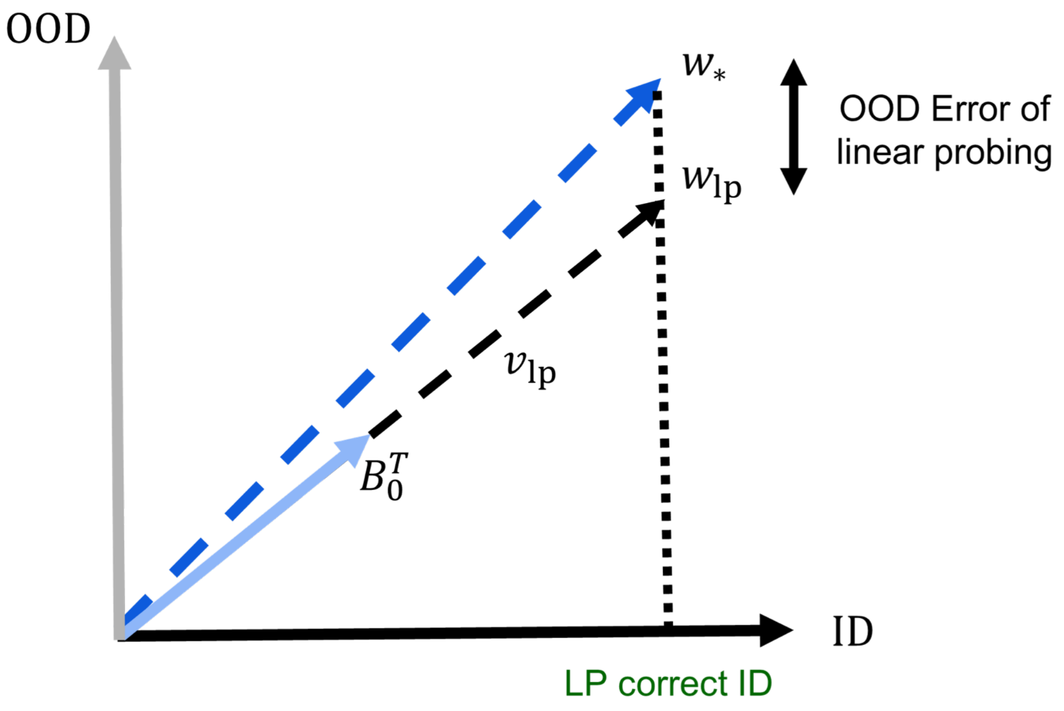


(a) Toy example (Linear probing)

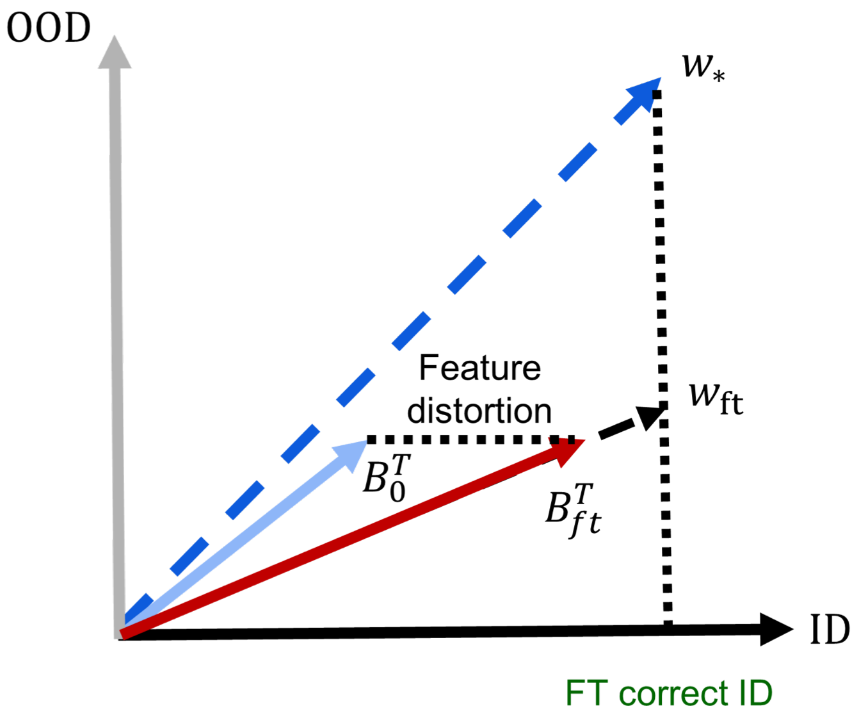


(b) Toy example (fine-tuning)

Feature Distortion (Toy Example)

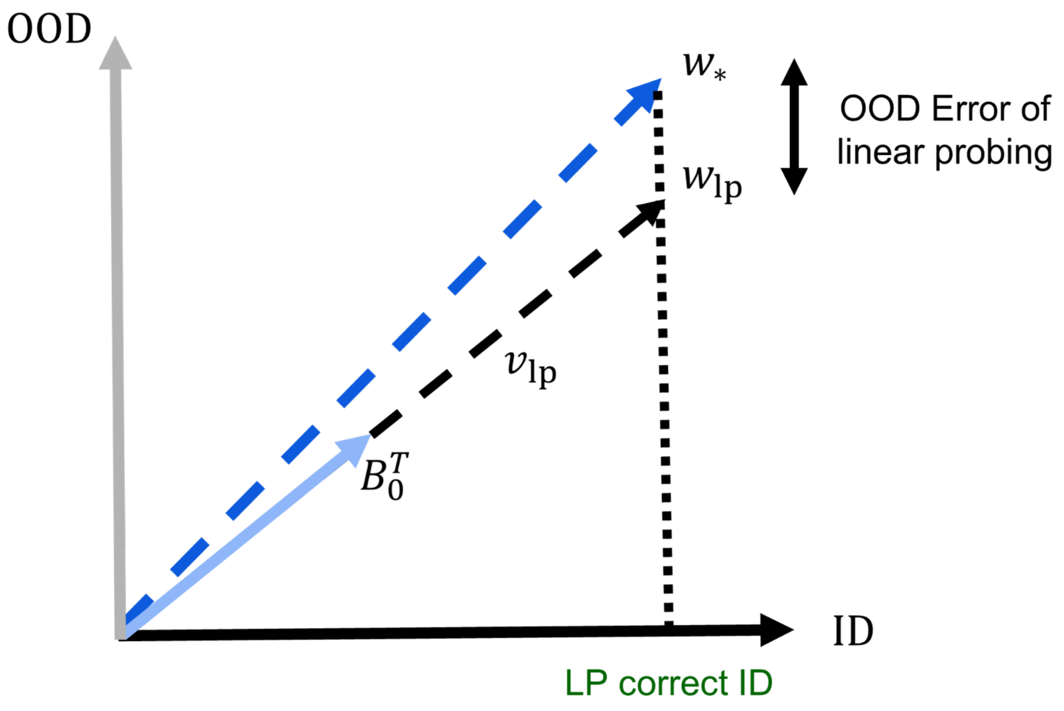


(a) Toy example (Linear probing)

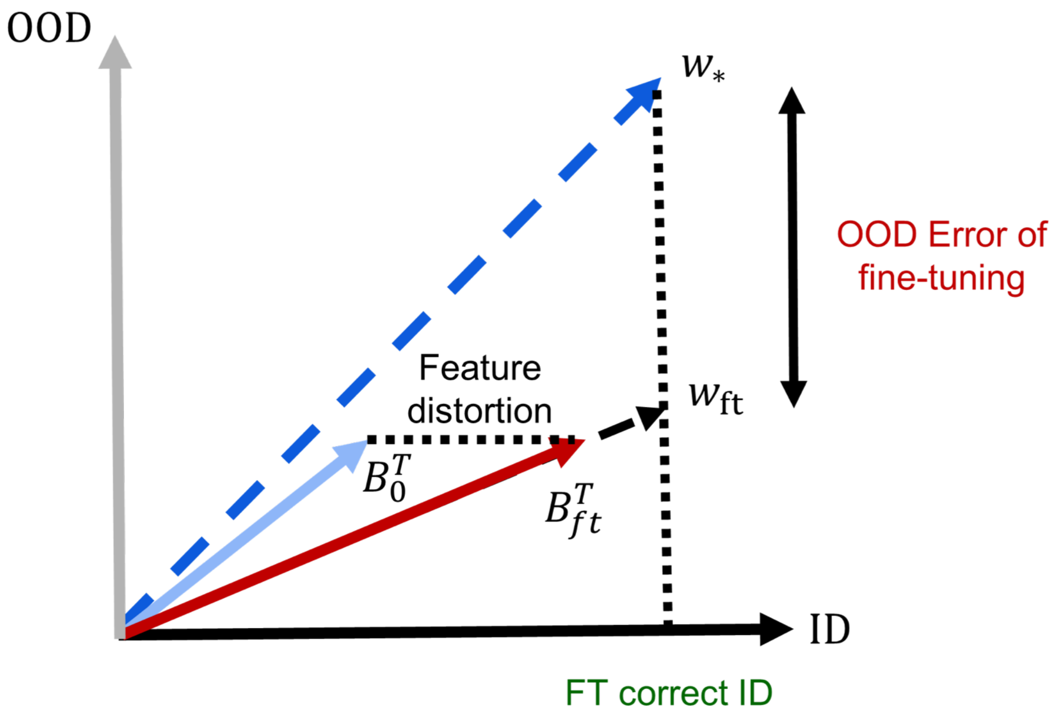


(b) Toy example (fine-tuning)

Feature Distortion (Toy Example)



(a) Toy example (Linear probing)



(b) Toy example (fine-tuning)

How to learn pretrained features

- Need to learn good features for *both* ID and OOD
- Auxiliary information
  - In-N-Out: Pre-Training and Self-Training using Auxiliary Information for Out-of-Distribution Robustness. SMX*, **AK***, RJ*, FK, TM, PL. ICLR 2021.
- Contrastive learning
  - Connect, Not Collapse: Explaining Contrastive Learning for Unsupervised Domain Adaptation. KS*, RJ*, **AK***, SMX*, JZH, TM, PL. ICML 2022 (Long Talk).

Outline

1. Fine-tuning can do worse than linear-probing OOD
2. Why fine-tuning can underperform OOD
3. **Simple change to fine-tuning: improved accuracy on 10 datasets**

Improving fine-tuning

- Fine-tuning works better on ID test; linear probing works better on OOD test
- Reason: start with random head, changes a lot → features get distorted

Can we refine features without distorting them too much?

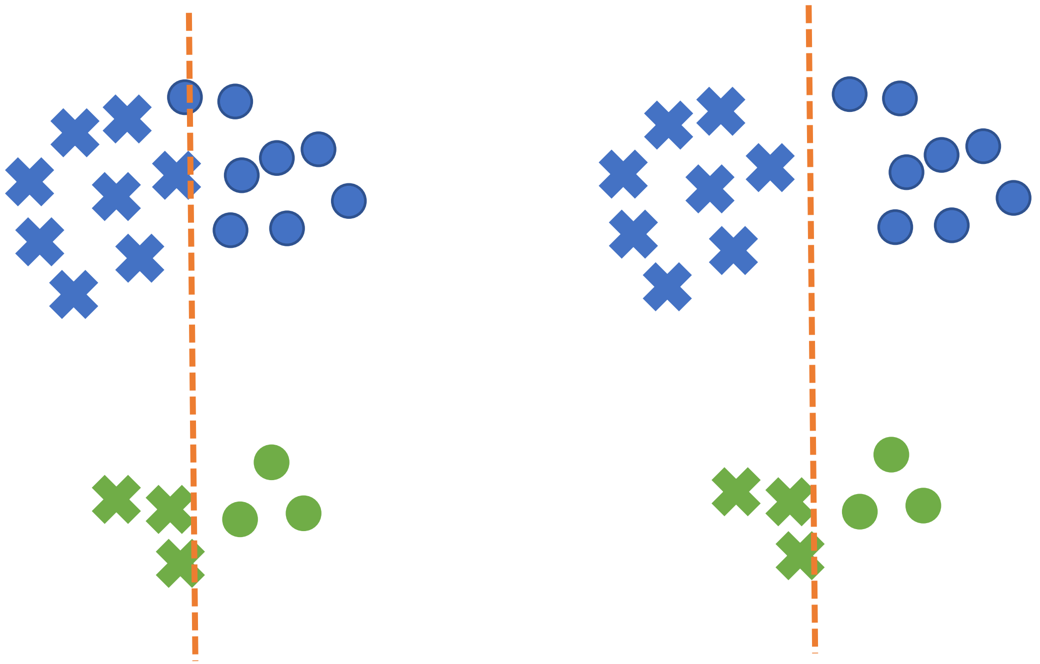
LP-FT

Step 1: Linear probe

Step 2: Fine-tune

(Levine et al 2016, Kanavati & Tsuneki, 2021)

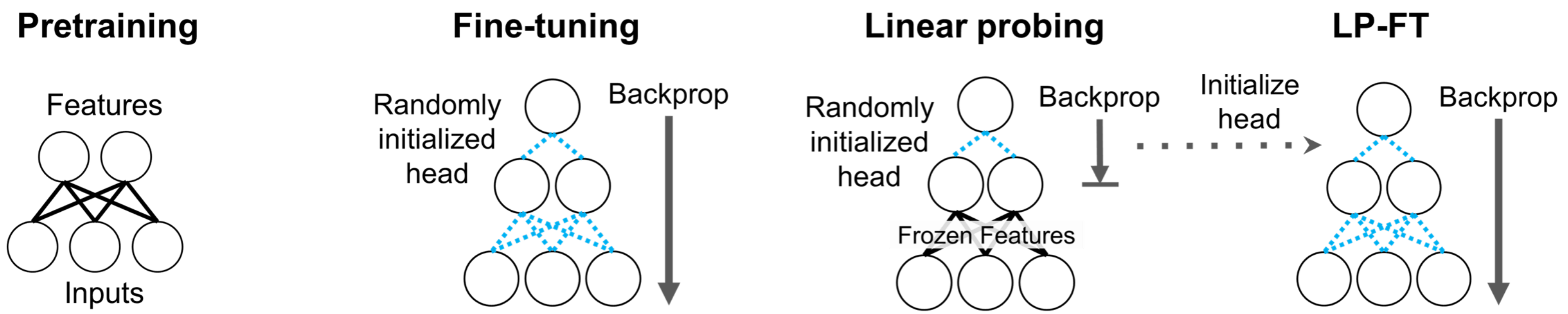
Prove this intuition in a simple setting



Improving fine-tuning: experiments

- Datasets: standard datasets including CIFAR, ImageNet, DomainNet, BREEDS, satellite remote sensing
- Models: conv nets (ResNet-50) and Vision Transformers (ViT-B/16)
- Protocols:
  - Rigorous protocol for tuning hyperparameters on ID validation data
  - Ensure that LP-FT and fine-tuning use the same computation

Improving fine-tuning



ID test

OOD test

ID test	85.1%	82.9%	**85.7%**
OOD test	59.3%	66.2%	**68.8%**

Average accuracies (10 datasets)

+10% over fine-tuning!

In-Distribution Accuracies

	CIFAR-10	Ent-30	Liv-17	DomainNet	FMoW	ImageNet	Average
FT	**97.3 (0.2)**	**93.6 (0.2)**	97.1 (0.2)	84.5 (0.6)	**56.5 (0.3)**	**81.7 (-)**	85.1
LP	91.8 (0.0)	90.6 (0.2)	96.5 (0.2)	89.4 (0.1)	49.1 (0.0)	79.7 (-)	82.9
LP-FT	**97.5 (0.1)**	**93.7 (0.1)**	**97.8 (0.2)**	**91.6 (0.0)**	51.8 (0.2)	**81.7 (-)**	**85.7**

Out-of-Distribution Accuracies

	STL	CIFAR-10.1	Ent-30	Liv-17	DomainNet	FMoW
FT	82.4 (0.4)	92.3 (0.4)	60.7 (0.2)	77.8 (0.7)	55.5 (2.2)	32.0 (3.5)
LP	85.1 (0.2)	82.7 (0.2)	**63.2 (1.3)**	82.2 (0.2)	79.7 (0.6)	**36.6 (0.0)**
LP-FT	**90.7 (0.3)**	**93.5 (0.1)**	**62.3 (0.9)**	**82.6 (0.3)**	**80.7 (0.9)**	**36.8 (1.3)**

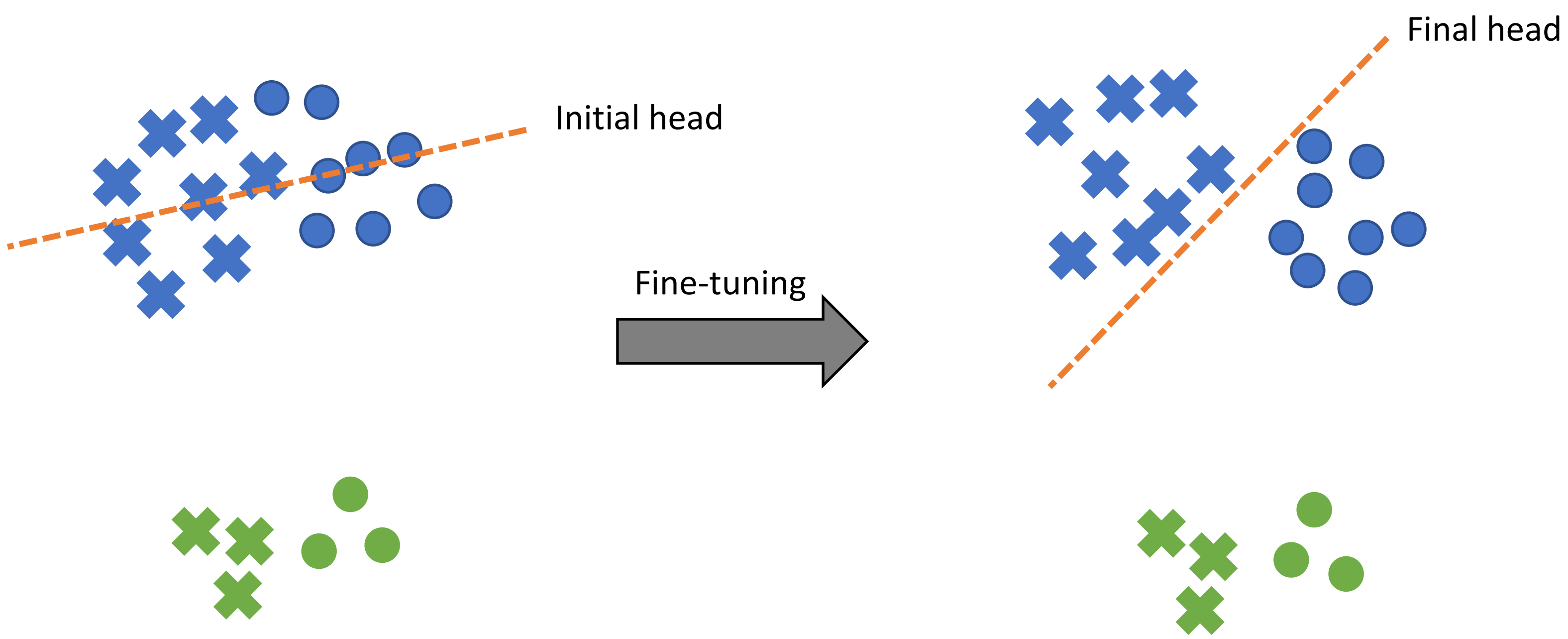
	ImNetV2	ImNet-R	ImNet-Sk	ImNet-A	Average
FT	**71.5 (-)**	52.4 (-)	40.5 (-)	27.8 (-)	59.3
LP	69.7 (-)	70.6 (-)	46.4 (-)	45.7 (-)	66.2
LP-FT	**71.6 (-)**	**72.9 (-)**	**48.4 (-)**	**49.1 (-)**	**68.9**

State-of-the-Art Accuracies

- Model Soups paper (Wortsman, …, Carmon*, Kornblith*, Schmidt*, 2022)
- Fine-tune ViT-G/14 (pretrained on JFT-3B) many times with LP-FT using different hyperparameters, average their weights in a greedy strategy (add a new model to the "soup" if ID validation accuracy improves)
- SoTA on ImageNet, ImageNet-(V2, Sketch, R, A), WILDS-iWildCam, WILDS-FMoW, and more

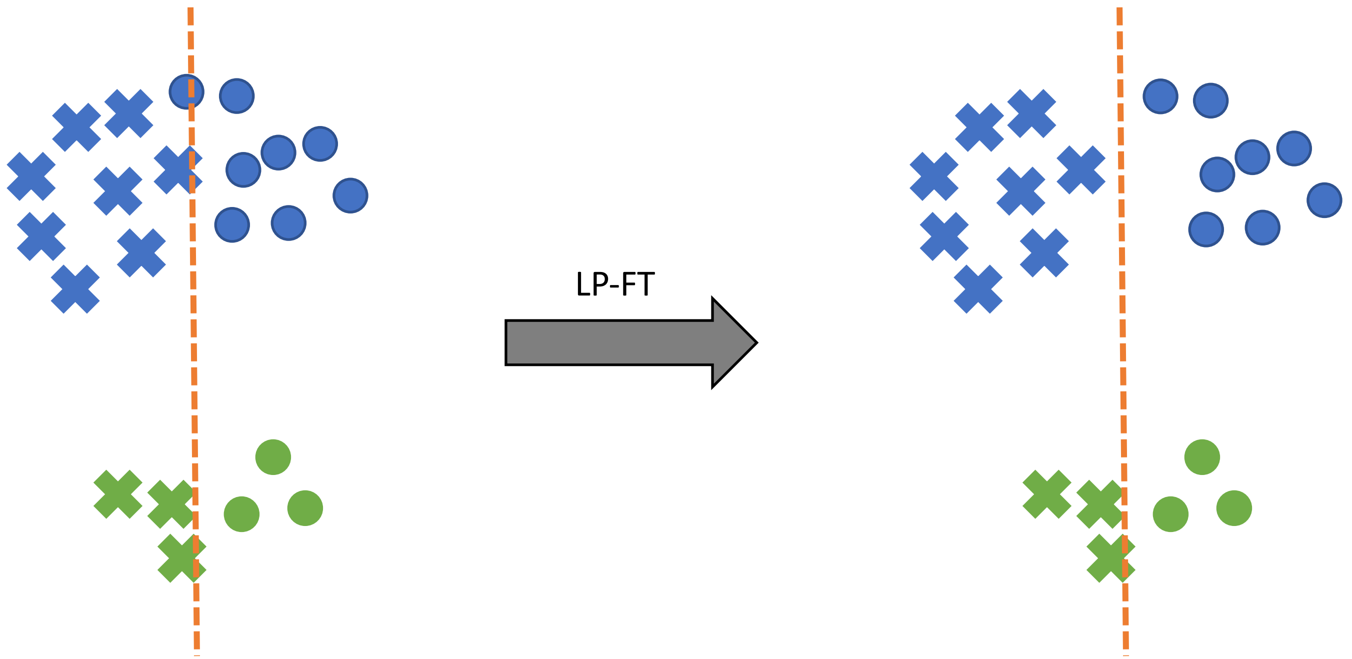
Does feature distortion happen?

- ID features change more than OOD features



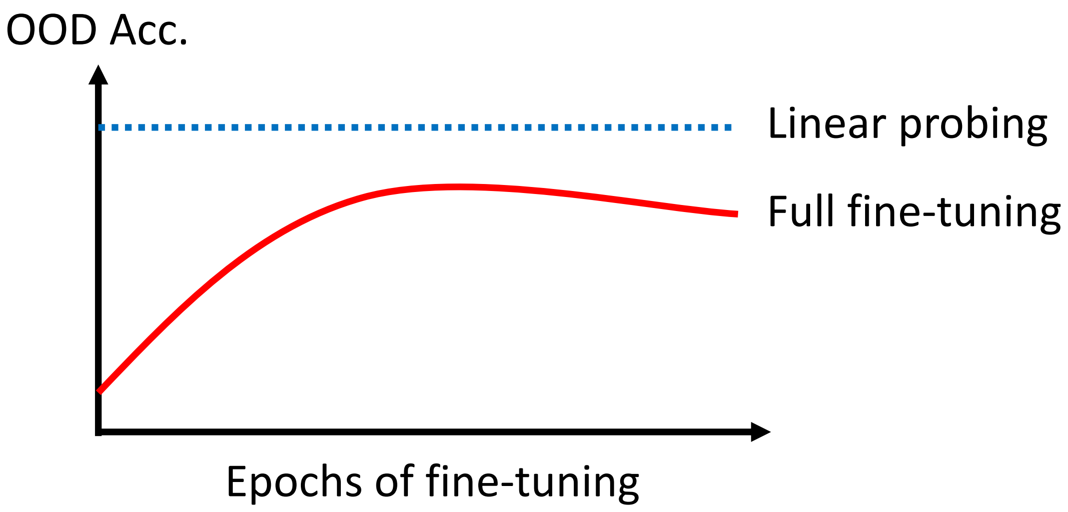
Does feature distortion happen?

- Features change orders of magnitude less with LP-FT



Does feature distortion happen?

- Early stopping does not solve the problem with fine-tuning



Important conditions for LP vs. FT

- Theory says fine-tuning does worse than linear probing **if** features good, distribution shift large
- CIFAR-10.1, ImageNetV2: small shift, FT does better
- Use MoCo-V1 instead of MoCo-V2: worse features, FT does better

Discussion

- Pretrained models give large improvements in accuracy, but how we fine-tune them is key
- LP-FT is just a starting point, better methods?
- What to do when linear probing not so good?

Discussion – Future Work

- Tighter analysis (including lower / upper bounds) for fine-tuning
- What happens for deep non-linear networks & classification?
- LP-FT analysis very toy, interaction with regularization?

Discussion - Related Work

- Lightweight fine-tuning
  - Can often improve OOD accuracy, we give one explanation
  - Increasingly important as pretrained feature quality improves
  - Adapter tuning, prefix tuning, composed fine-tuning
- Linear probing then fine-tuning
  - Sometimes used as a heuristic for ID, e.g. ULMFit
  - Just a starting point

Summary

1. Fine-tuning can do worse than linear-probing OOD
2. Why fine-tuning can underperform OOD
3. Simple change to fine-tuning: improved accuracy on 10 datasets
    1. Linear probe to learn good head initialization
    2. Fine-tune to refine features

Appendix: Few-Shot vs. OOD

- Result lower bounds error of fine-tuning, whenever test data contains directions outside training span
- This happens if:
  - Standard IID setting, when we have very few training examples
  - Distribution shift, no matter the number of training examples

Appendix: Regularization vs LP-FT

- Compared LP-FT with many other methods on Living-17, including regularizing towards pretrained weights, higher learning rate for top layer, side-tuning---LP-FT did better
- Regularization: suspect its an optimization explanation, with a random head the weights change initially, and end up at different part of loss landscape?
- 2-layer linear networks: regularization makes some local minima bad