

Freeze then Train: Towards Provable Representation Learning under Spurious Correlations and Feature Noise





Haotian Ye, James Zou, Linjun Zhang

Peking University, Stanford University, Rutgers University

AISTATS 2023

- 1 Introduction
- 2 Understanding LLR
- 3 Improving LLR Performance
- 4 Experiments on Common Benchmarks
- 5 Conclusion

Spurious Correlations

CelebA hair color dataset				
	\mathcal{G}_1	\mathcal{G}_2	\mathcal{G}_3	\mathcal{G}_4
Image Examples				
Description	Non-blond woman	Non-blond man	Blond woman	Blond man
Class Label	0	0	1	1
# Train data	71629 (44%)	66874 (41%)	22880 (14%)	1387 (1%)
# Val data	8535	8276	2874	182
Target: hair color;	Spurious feature: gender;		Minority: \mathcal{G}_4	

- Real-world datasets are riddled with spurious correlations.
- The goal is to learn a model in training environments with spurious correlations, such that it performs well in test environments where the correlations are broken.
- This is challenging as we do not explicitly know which features are spurious and which are core.

A New Strategy: Last Layer Retraining

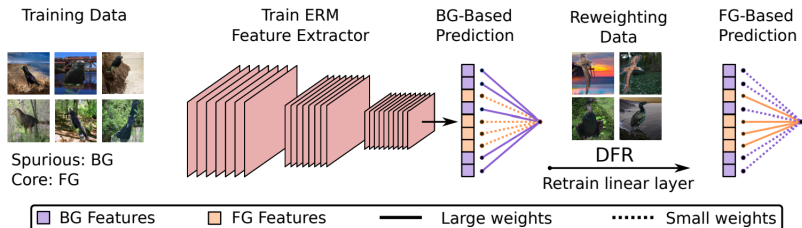
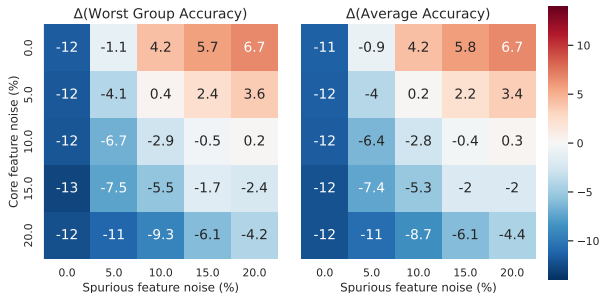


Figure: LLR pipeline from Kirichenko et al. (2022).

- Recently, it has been found that ERM still learns the core features under spurious correlations, despite its test performance being bad.
- A new strategy “last layer retraining” (LLR): learn features in training environments, and retrain the last layer (linear probe) in test environments.
 - Less demanding and more effective than OOD generalization;
 - More applicable and computationally efficient than general domain adaptation.
 - **However, this understanding is incomplete.**

Our Motivation: When and Why LLR works

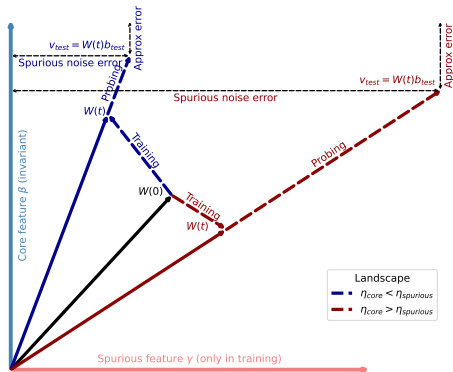


- We empirically found that ERM **only** learns the core features (thus performing well in LLR) **when** the noise of core features is much smaller than that of spurious features.
- Our goal: understand **when and why** the core features can and cannot be learned during training and recovered after retraining the last layer.
 - Why is noise so important for LLR performance?
 - How to make LLR work again under different noise conditions?

- 1 Introduction
- 2 Understanding LLR**
- 3 Improving LLR Performance
- 4 Experiments on Common Benchmarks
- 5 Conclusion

Intuition: Why Noises Matter

- LLR performance is determined by the quality of the learned features.
- ERM typically learns a mixture of different features.



- Importantly, the proportion depends on the trade-off between information and noise: features with larger noise are used less.
- During LLR, when the proportion of the core feature is small, we suffer more to amplify this feature.
- In this illustrative figure, **the red curve** induces a large LLR error than **the blue curve**.
- We formalize this into the next theorems.

A Minimum Theoretical Framework

Consider a two-layer non-convex optimization problem:

- Model: $f(\mathbf{x}) = \mathbf{x}\mathbf{W}\mathbf{b}$, where $\mathbf{W} \in \mathbb{R}^{d \times m}$ is the feature learner and $\mathbf{b} \in \mathbb{R}^{m \times 1}$ is the last layer.
- Data: assume the data (\mathbf{x}, y) is generated from the following mechanism:

$$\mathbf{x}_1 \sim \mathbb{P} \in \Delta(\mathbb{R}^{1 \times d_1}), y = \mathbf{x}_1 \beta + \epsilon_{core},$$

$$\mathbf{x}_2 = \begin{cases} y\gamma^\top + \epsilon_{spu} & \text{Training} \\ \epsilon_{spu} & \text{Testing} \end{cases} \in \mathbb{R}^{1 \times d_2}, \mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \in \mathbb{R}^{1 \times d}.$$

Here $\epsilon_{core} \in \mathbb{R}, \epsilon_{spu} \in \mathbb{R}^{1 \times d_2}$ are independent core and spurious noises with mean zero and variance (covariance matrix) η_{core}^2 and η_{spu}^2 respectively.

- Optimization:
 - Training stage: $\ell_{tr}(\mathbf{W}, \mathbf{b}) = \frac{1}{2n} \sum_{i=1}^n (f(\mathbf{x}_i) - y_i)^2$ (gradient flow).
 - Testing stage: $\ell_{te}(\mathbf{W}) = \min_{\mathbf{b}} \mathbb{E}_{(\mathbf{x}, y) \sim \mathbb{D}_{te}} \frac{1}{2} (f(\mathbf{x}) - y)^2$, where \mathbf{W} is from the training stage, and \mathbf{b} can be obtained using standard LR techniques.

Theorems: Noises Matter

$\eta_{core} < \eta_{spu}$ (Informal upper bound)

Under some assumptions, for any $\eta_{core} < \eta_{spu}$, any time t ,

$$\ell_{te}(\mathbf{W}(t)) \leq \left(1 + \frac{\eta_{core}^2}{\eta_{spu}^2}\right) \text{err}_{te}^* + \mathcal{O}(t^{-1}),$$

where err_{te}^* is the optimal testing error.

$\eta_{core} > \eta_{spu}$ (Informal lower bound)

Under some assumptions, for any $\eta_{core} > \eta_{spu}$,

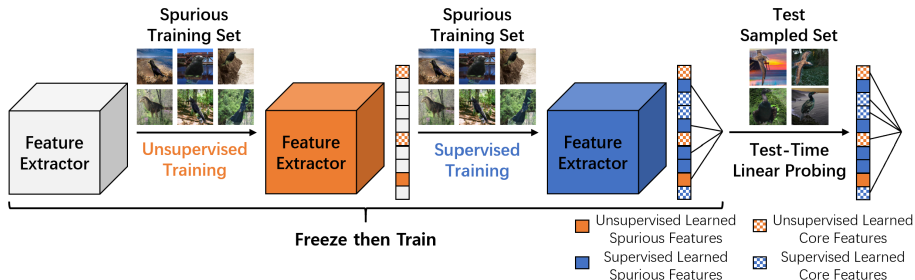
$$\lim_{t \rightarrow \infty} \frac{\ell_{te}(\mathbf{W}(t))}{\text{err}_{te}^*} \geq 1 + \frac{\eta_{core}^2}{2\eta_{spu}^2} \left(1 \wedge \frac{1}{2\eta_{spu}^2 \|\Sigma^{-1}\|_2 \|\mathbf{W}_1^\dagger(\infty)\|_2^2}\right),$$

where \mathbf{W}^\dagger is the Moore-Penrose inverse, and $a \wedge b$ takes the minimum over a, b .

- 1 Introduction
- 2 Understanding LLR
- 3 Improving LLR Performance**
- 4 Experiments on Common Benchmarks
- 5 Conclusion

Algorithm: Freeze then Train (FTT)

- Takeaway from theorems: features learned in a supervised way are sometimes biased under certain noise conditions.
- Intuition: we should also learn features *in an unsupervised way*.
 - Noise in the label mechanism no longer influences learning.
 - Preserve features that are useful in the testing stage but are ruled out because they are less informative than other features w.r.t. labels.
- Algorithm FTT: freezes certain salient features unsupervisedly and then trains the rest of the features supervisedly.



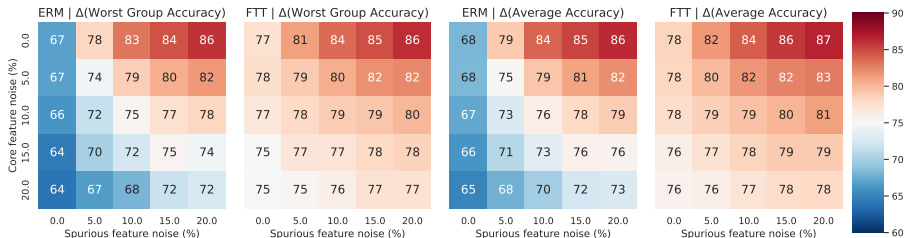
Properties of FTT

FTT Bound (Informal)

Under some assumptions, for almost any $\eta_{core}, \eta_{spu} > 0$, any time t ,

$$\ell_{te}(\mathbf{W}_{FTT}(t)) \leq \text{err}_{te}^* + \mathcal{O}(t^{-1}).$$

Compared with ERM, FTT can learn core features even when $\eta_{core} > \eta_{spu}$, and achieves good LLR performance.



- 1 Introduction
- 2 Understanding LLR
- 3 Improving LLR Performance
- 4 Experiments on Common Benchmarks**
- 5 Conclusion

Spurious Correlation: Waterbirds and CelebA

- We compare the LLR performance of FTT with ERM, IRM, CVaR-DRO and JTT on Waterbirds and CelebA, two popular spurious correlation benchmarks.
- We explicitly generate noise by flipping labels, with spurious noise equaling 5% and core noise ranging from 0% to 10%.

Dataset	η_{core} (%)	Worst Group Accuracy (%)					Average Accuracy (%)				
		ERM	IRM	CVaR-DRO	JTT	Ours	ERM	IRM	CVaR-DRO	JTT	Ours
Waterbirds	0	95.0	95.3	94.3	93.3	94.5	95.3	95.5	94.6	94.1	94.9
	2	93.6	94.1	93.8	89.7	93.6	94.2	94.3	94.0	90.7	94.2
	4	92.8	92.8	92.8	85.3	92.9	93.2	93.5	93.2	85.9	93.5
	6	90.8	91.5	77.8	86.8	92.8	91.3	91.8	77.8	87.1	92.9
	8	88.5	88.8	77.8	82.0	92.7	89.9	90.1	77.8	82.7	93.0
	10	87.6	87.9	77.8	78.6	92.4	89.4	89.4	77.8	78.9	92.9
	Mean	91.4	91.7	85.7	86.0	93.1	92.2	92.4	85.9	86.6	93.6
CelebA	0	95.0	95.2	92.9	94.4	95.3	97.2	97.2	96.0	96.7	97.2
	2	95.2	95.2	92.4	91.6	95.2	97.2	97.2	95.9	96.0	97.2
	4	94.5	94.2	91.9	92.7	94.9	97.1	97.0	95.5	96.4	97.2
	6	94.3	94.3	91.5	92.0	94.4	96.9	96.9	95.5	96.0	97.0
	8	93.7	93.8	91.4	91.4	94.0	96.7	96.7	95.4	95.7	96.7
	10	92.4	92.8	91.1	80.5	93.1	96.2	96.2	95.4	92.1	96.3
	Mean	94.2	94.2	91.9	90.4	94.5	96.9	96.9	95.6	95.5	96.9

Table 2: Test-time probing accuracy (%) for four methods on Waterbirds and CelebA, under different core noises η_{core} . **Bold** means the best accuracy across four methods. The “Mean” row stands for the average accuracy across η_{core} . We repeat all settings 10 times and average the numbers. For worst group accuracy, FTT (ours) can be competitive when η_{core} is small and outperform other algorithms by at most 4.5% when η_{core} increases. It can increase accuracy by 1.4% and 0.3% on Waterbirds and CelebA on average.

Distribution Shift: VLCS, PACS, and Office-Home

- We further study FTT in more general distribution shift benchmarks, where there is no explicit spurious correlation and noise between features and labels.
- We consider three OOD multi-class classification datasets: PACS, VLCS, and Office-Home.
- We compare the LLR accuracy of FTT, ERM, IRM, and Group-DRO.
- FTT is initially designed for spurious correlation benchmarks. However, we find that it also works well in general OOD problems.

	Domain	A	C	P	S	Mean
PACS	ERM	89.2	93.2	95.8	88.4	91.7
	IRM	61.1	67.5	81.7	79.1	72.4
	DRO	91.9	92.7	95.8	91.3	93.0
	Ours	92.7	94.9	97.9	90.8	94.1
	Domain	A	C	P	R	Mean
Office-Home	ERM	69.9	69.9	87.8	78.8	76.6
	IRM	25.2	44.9	69.3	54.0	48.3
	DRO	72.8	73.1	88.5	79.9	78.6
	Ours	73.8	73.7	87.1	83.1	79.4
	Domain	C	L	S	V	Mean
VLCS	ERM	99.3	75.0	77.3	81.5	83.3
	IRM	75.3	62.5	59.5	60.4	64.4
	DRO	99.6	74.0	78.5	81.8	83.5
	Ours	100.0	76.6	81.1	84.6	85.6

Table 3: Test-time probing accuracy (%) for 4 methods on PACS, Office-Home, and VLCS. Rows “Domain” specify which domain among 4 domains is unseen during the training stage, therefore used to retrain the last layer. The “Mean” column stands for the average accuracy across different test domain selections, and we **bold** the highest accuracy among 4 methods in each setting. FTT (ours) consistently outperforms other methods by 1.1% on PACS, 0.8% on Office-Home, and 2.1% on VLCS.

- 1 Introduction
- 2 Understanding LLR
- 3 Improving LLR Performance
- 4 Experiments on Common Benchmarks
- 5 Conclusion**

Takeaway

- We empirically show that ERM only learns the core features under certain noise conditions.
- We give a theoretical framework to explain this phenomenon. We point out that noise is an essential factor, because labels will incentivize the ERM model to use features with smaller noise.
- To improve LLR performance, we propose FTT that combines supervised learning and unsupervised learning.
- Under spurious correlation and distribution shift benchmarks, FTT consistently outperforms other purely supervised algorithms.
- Our code can be found [here](#). Hope you like our work!

I will be starting CS PhD at Stanford this Fall. Please feel free to contact me if you find my work interesting! [My homepage](#).

Thanks!