OUT-OF-DISTRIBUTION GENERALIZATION ANALYSIS VIA INFLUENCE FUNCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

The mismatch between training dataset and target environment is one major challenge for current machine learning systems. When training data is collected from multiple environments and the the evaluation is on any new environment, we are facing an Out-of-Distribution (OOD) generalization problem that aims to find a model with the best OOD accuracy, i.e. the best worst-environment accuracy. However, with limited access to environments, the worst environment may be unseen, and test accuracy is a biased estimate of OOD accuracy. In this paper, we show that test accuracy may dramatically fail to identify OOD accuracy and mislead the tuning procedure. To this end, we introduce Influence Function, a classical tool from robust statistics, into the OOD generalization problem and suggest the variance of influence function to measure the stability of a model on training environments. We show that the proposed index and test accuracy together can help us discern whether OOD algorithms are needed and whether a model achieves good OOD generalization.

1 INTRODUCTION

Most machine learning systems assume that both training and test data are independently and identically distributed (IID), which does not always hold in practice (Bengio et al. (2019)). Consequently, its performance is often greatly degraded when the test data is from a different environment (distribution). A classical example is the problem to identify cows and camels (Beery et al. (2018)), where the empirical risk minimization (ERM, Vapnik (1992)) may classify images by background color instead of object shape. As a result, when the test environment is "out-of-distribution" (OOD), e.g. when the background color is changed, its performance will drop significantly. To achieve OOD generalization is to obtain a robust predictor against this distribution shift.

Suppose that we have training data under m environments:

$$\mathbb{S} = \{\mathbb{S}^e : e \in \mathcal{E}_{tr}, |\mathcal{E}_{tr}| = m\}, \quad \mathbb{S}^e = \{\boldsymbol{z}_1^e, \boldsymbol{z}_2^e, \ldots, \boldsymbol{z}_{n^e}^e\} \text{ with } \boldsymbol{z}_i^e \sim P^e, \tag{1}$$

where P^e is the data distribution corresponding to environment e, $\mathcal{E}_{tr}$ is the set of all training environments, and each $\boldsymbol{z}_i^e$ is a data point. The out-of-distribution problem is to find a model f_{OOD} such that

$$f_{\text{OOD}} = \arg\min_f \sup_{e \in \mathcal{E}_{all}} \ell(f, P^e), \tag{2}$$

where $\mathcal{E}_{all}$ is the set of all possible environments and $\ell(f, P^e)$ is the expected loss of f on environment e. Motivated by understanding from causal view, recent algorithms address this OOD problem by recovering causal features and build the optimal model on top of these features, such as Invariant Risk Minimization (IRM, Arjovsky et al. (2019)) and Risk Extrapolation (REx, Krueger et al. (2020)). Most works evaluate on Colored MNIST (see 5.2 for details) where it makes explicit which features are causal and invariant and which are spurious and non-causal, allowing to directly obtain OOD accuracy. Here, OOD accuracy stands for the worst environment accuracy on $\mathcal{E}_{all}$. On the other hand, the OOD algorithms cannot outperform the SOTA method in some domain generalization and domain adaptation tasks, e.g. VLCS (Torralba & Efros (2011)) and PACS (Li et al. (2017)). This is not surprising, since these tasks only require high performance on certain environment, while an OOD algorithm is expected to learn truly invariant features and be excellent on $\mathcal{E}_{all}$. This phenomenon is described as "accuracy-vs-invariance trade-off" in Akuzawa et al. (2019).

Two questions arise in problem (2). First, previous works assume that the distributions $P^e \in \mathcal{E}_{all}$ are different. But in reality, we do not know whether it is true. If not, slavishly implementing OOD algorithms may hurt the performance and also waste time. Second, given two candidate models, which model achieves a better OOD property? Furthermore, how to discern a model's OOD property? Traditionally, test accuracy is used for the judgement: higher test accuracy means better OOD performance. However, under most circumstances, available environments is a subset of $\mathcal{E}_{all}$, so OOD accuracy may be underestimated. More seriously, test accuracy may be irrelevant or even *inversely correlated* to OOD accuracy, i.e. a model with better OOD property achieves lower test accuracy. This phenomenon is not uncommon, especially when there are features that are virtually spurious in $\mathcal{E}_{all}$ but show a strong correlation to the target in $\mathcal{E}_{tr}$, i.e. background color in cows and camels classification.

We briefly give an example that test accuracy misleads us and leave further analysis to Appendix A.3. Consider three environments with $p^e \in \{0.0, 0.1, 0.2\}$ in Colored MNIST (for meaning of p^e, please refer to 5.2). We choose any two environments to be training environments and the rest to be the test one. In each of three setups, we run IRM and ERM 100 times each and record their test accuracy as well as real OOD accuracy (Figure 1). No matter which environment is used as the test one, ERM model uniformly outperforms IRM model on test accuracy. However, IRM model achieves consistently better OOD accuracy. Shortcomings of test accuracy here is obvious. The naive use of test accuracy may lead to a non-OOD model being chosen as the best.

Motivated by the obstacles above, we hope to find an index that correlates better with model's OOD property, *even when accessible environments are limited and "worst" remains unknown*. The problem of test accuracy is that it pays no attention to what model has learnt, but considers only the accuracy on a specific environment. Therefore, we need to focus on what features the model has learnt (hope the causal, invariant one) and avoided (hope the non-causal, spurious one). Specifically, we want to measure the extent to which the model learns features that are invariant across environments, and Influence function (Cook & Weisberg (1980)) can serve our purpose. Influence function was proposed to measures the parameter change when a data point is removed or upweighted by a small perturbation (details in 3.2). When modified it to environmental level, it measures the influence of an environment instead of a data point on the model. Base on this, the variance of influence function allows us to measure OOD property and solve the obstacles.

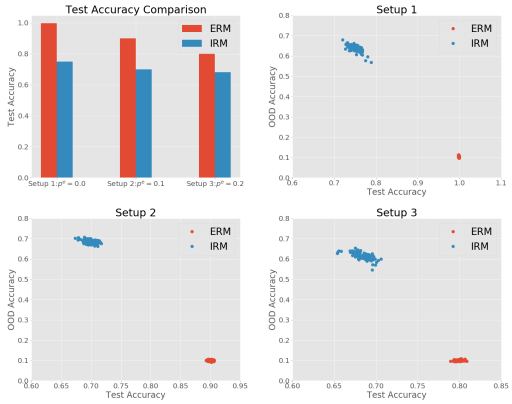

Figure 1: Experiments in Colored MNIST to show test accuracy is not enough to reflect a model's OOD accuracy. The top left penal shows the test accuracy of ERM and IRM. The other three panels present the relationship between test accuracy (x-axis) and OOD accuracy (y-axis) in three setups.

We summarize our contributions as follows:

- We introduce influence function to environmental level and propose index $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ (formula 5) based on environmental influence function of the model $f_{\boldsymbol{\theta}}$. Our index can measure the OOD extent of available environments, i.e. how different these environments (distributions) are. This measurement provides a basis for whether to adopt an OOD algorithm and to collect more diverse environments. See Section 4.1 and Section 5.2.1 for details.

- We point out that the proposed index $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ can solve the weakness of test accuracy. Specifically, under most OOD generalization problems, using test accuracy and our index together, we can discern the OOD property of a model. See Section 4.2 for details.

- We propose to use only a small but important part of the model to calculate the influence function. This overcomes the huge computation cost of solving the inverse of Hessian. It is not merely for calculation efficiency and accuracy, but it coincides with our understanding that only these parameters capture what features a model has learnt. See section 4.3 for details.

We organize our paper as follows: Section 2 reviews related works and Section 3 introduces the preliminaries of OOD methods and influence function. Section 4 presents our proposal and detailed analysis. Section 5 shows various experiments. The conclusion is given in Section 6.

2 RELATED WORK

The mismatch between the development dataset and the target environment is one major challenge in machine learning (Castro et al. (2020); Kuang et al. (2020)). Many works assume that the ground truth can be represented by a causal Direct Acyclic Graph (DAG), and they use the DAG structure to discuss the worst-environment performance (Rojas-Carulla et al. (2018); Peters et al. (2016); Subbaswamy et al. (2019); Bühlmann et al. (2020); Magliacane et al. (2018)). All these works employ multiple environment data and causal assumptions to discover the parents of the target variable. Rojas-Carulla et al. (2018) and Magliacane et al. (2018) also apply this idea to Domain Generalization and Multi-Task Learning setting. Starting from multiple environment data rather than model assumptions, Arjovsky et al. (2019) proposes Invariant Risk Minimization (IRM) to extract causal (invariant) features and learn invariant optimal predictor on the top of the causal features. It analyzes the generalization properties of IRM from the view of sufficient dimension reduction (Cook (2009); Cook et al. (2002)). Ahuja et al. (2020) considers IRM as finding the Nash equilibrium of an ensemble game among several environments and develops a simple and stable training algorithm. Krueger et al. (2020) derives the Risk Extrapolation (REx) to extract invariant features and further derives a practical objective function via variance penalization. Xie et al. (2020) employs a framework from distributional robustness to interpret the benefit of REx comparing to robust optimization (Ben-Tal et al. (2009); Bagnell (2005)). On the other hand, Adversarial Domain Adaption (Li et al. (2018); Koyama & Yamaguchi (2020)) uses discriminator to look for features that are independent of environment and uses these features for further prediction.

Influence function is a classic method from the robust statistics literature (Robins et al. (2008; 2017); Van der Laan et al. (2003); Tsiatis (2007)). It can be used to track the impact of a training sample on the prediction. Koh & Liang (2017) proposes a second-order optimization technique to approximate the influence function. They verify their method with different assumptions on the empirical risk ranging from being strictly convex and twice-differentiable to non-convex and non-differentiable losses. Koh et al. (2019) also estimates the effect of removing a subgroup of training points via influence function. They find out that the approximation computed by the influence function is correlated with the actual effect. Influence function has been used in many machine learning tasks. Cheng et al. (2019) proposes an explanation method, Fast Influence Analysis, that employs influence function on Latent Factor Model to solve the lack of interpretability of the collaborative filtering approaches for recommender systems. Cohen et al. (2020) uses influence function to detect adversarial attacks. Ting & Brochu (2018) proposes an asymptotically optimal sampling method via an asymptotically linear estimator and the associated influence function. Alaa & Van Der Schaar (2019) develops a model validation procedure that estimates the estimation error of causal inference methods. Besides, Fang et al. (2020) leverages influence function to select a subset of normal users who are influential to the recommendations.

3 PRELIMINARIES

3.1 ERM, IRM AND REX

In this section, we give some notations and introduce the OOD methods we use. Recall the multiple environment setup (1) and OOD problem (2). For an environment P^e and a hypothetical model f, the population loss is $\ell(f, P^e) = \mathbb{E}_{\mathbf{z}\sim P^e}[L(f, \mathbf{z})]$ where $L(f, \mathbf{z})$ is the loss function on $\mathbf{z}$. The

empirical loss, which is the objective of ERM, is given by

$$\ell(f,\mathbb{S}) = \frac{1}{m}\sum_{e\in\mathcal{E}_{tr}} \ell(f,\mathbb{S}^e) \quad \text{with} \quad \ell(f,\mathbb{S}^e) = \frac{1}{n}\sum_{i=1}^{n} L(f,\boldsymbol{z}_i^e).$$

Recent OOD methods propose some novel regularized objective functions in the form:

$$\mathcal{L}(f,\mathbb{S}) = \ell(f,\mathbb{S}) + \lambda R(f,\mathbb{S}) \tag{3}$$

to discover f_{OOD} in (2). Here $R(f,\mathbb{S})$ is a regularization term and λ is the tuning parameter which controls the degree of penalty. Note that ERM is a special case by setting $\lambda = 0$. For simplicity, we will use $\mathcal{L}(f,\mathbb{S})$ to represent the total loss in case of no ambiguity. Arjovsky et al. (2019) focuses on the stability of f_{OOD} and considers the regularization:

$$R(f,\mathbb{S}) = \sum_{e\in\mathcal{E}_{tr}} \|\nabla_w \ell\big(g(w\Phi),\mathbb{S}^e\big)\big|_{w=1.0}\|^2$$

where Φ is the embedding mapping and g is the top model such that $f(\boldsymbol{x}) = g(\Phi(\boldsymbol{x}))$. Intuitively, $\Phi(\cdot)$ extracts features, and $g(\cdot)$ learns prediction rule on basis of these features. On the other hand, Krueger et al. (2020) encourages the uniform performance of f_{OOD} and proposes

$$R(f,\mathbb{S}) = \sum_{e\in\mathcal{E}_{tr}} (\ell(f,\mathbb{S}^e) - \ell(f,\mathbb{S}))^2.$$

3.2 Influence function and group effect

Consider a parametric hypothesis $f = f_{\boldsymbol{\theta}}$ and the corresponding solution: $\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}},\mathbb{S})$. By a quadratic approximation of $\mathcal{L}(f_{\boldsymbol{\theta}},\mathbb{S})$ around $\hat{\boldsymbol{\theta}}$, the influence function takes the form

$$\mathcal{IF}(\hat{\boldsymbol{\theta}},\boldsymbol{z}) = -\boldsymbol{H}_{\hat{\boldsymbol{\theta}}}^{-1}\nabla_{\boldsymbol{\theta}} L(f_{\hat{\boldsymbol{\theta}}},\boldsymbol{z}) \quad \text{with} \quad \boldsymbol{H}_{\hat{\boldsymbol{\theta}}} = \nabla^2_{\boldsymbol{\theta}}\mathcal{L}(f_{\hat{\boldsymbol{\theta}}},\mathbb{S}).$$

When the sample size of $\mathbb{S}$ is sufficiently large, the parameter change due to removing a data point z can be approximated by $-\mathcal{I}(z)/\sum_{e\in\mathcal{E}_{tr}}|\mathbb{S}^e|$ without retraining the model. Here $|\mathbb{S}^e| = n^e$ stands for the cardinal of the set $\mathbb{S}^e$. Furthermore, Koh et al. (2019) shows that the influence function can also predict the effects of large groups of training points (i.e. $\mathcal{Z} = \{z_1,...,z_k\}$), although there are significant changes in the model. The parameter change due to removing the group can be approximated by

$$\mathcal{IF}(\hat{\boldsymbol{\theta}},\mathcal{Z}) = -\boldsymbol{H}_{\hat{\boldsymbol{\theta}}}^{-1}\nabla_{\boldsymbol{\theta}}\frac{1}{|\mathcal{Z}|}\sum_{z\in\mathcal{Z}} L(f_{\hat{\boldsymbol{\theta}}},z).$$

Motivated by the work of Koh et al. (2019), we introduce influence function to OOD problem to address our obstacles.

4 Methodology

4.1 Influence of Environments

Following Arjovsky et al. (2019), we decompose a parametric hypothesis $f_{\boldsymbol{\theta}}$ into a top model and a feature extractor, i.e. $g(\Phi) = g(\Phi,\boldsymbol{\gamma})$, $\Phi(\boldsymbol{x}) = \Phi(\boldsymbol{x},\boldsymbol{\beta})$ and $\boldsymbol{\theta} = (\boldsymbol{\gamma},\boldsymbol{\beta})$. When upweighting an environment e by a small perturbation δ, we do not upweight the regularized term, i.e.

$$\mathcal{L}_+(\boldsymbol{\theta},\mathbb{S},\delta) = \mathcal{L}(\boldsymbol{\theta},\mathbb{S}) + \delta\cdot\ell(f,\mathbb{S}^e),$$

since the stability across different environments, which is encouraged by the regularization, should not depend on the sample size of an environment. Given a model $f_{\hat{\boldsymbol{\theta}}}$ learnt with loss function $\mathcal{L}(\boldsymbol{\theta},\mathbb{S})$ in formula 3, according to detailed derivation in Appendix A.2, the change of top model g caused by upweighting the environment is

$$\mathcal{IF}(\hat{\boldsymbol{\gamma}},\mathbb{S}^e|\hat{\boldsymbol{\theta}}) := \lim_{\delta\to 0^+}\frac{\Delta\boldsymbol{\theta}}{\delta} = -\boldsymbol{H}_{\hat{\boldsymbol{\gamma}}}^{-1}\nabla_{\boldsymbol{\gamma}}\ell(f_{\hat{\boldsymbol{\theta}}},\mathbb{S}^e), \quad e\in\mathcal{E}_{tr}. \tag{4}$$

Here $\boldsymbol{H}_{\hat{\boldsymbol{\gamma}}} = \nabla^2_{\hat{\boldsymbol{\gamma}}}\mathcal{L}(f_{\hat{\boldsymbol{\theta}}},\mathbb{S})$, and we assume that $\mathcal{L}$ is twice-differentiable in $\boldsymbol{\gamma}$. For a regularized method, e.g. IRM and REx, the influence of their regularized term is reflected in $\boldsymbol{H}$ and in learnt model $f_{\hat{\boldsymbol{\theta}}}$. As mentioned above, $\mathcal{IF}(\hat{\boldsymbol{\gamma}},\mathbb{S}^e|\hat{\boldsymbol{\theta}})$ measures change of model caused by upweighting environment e. Therefore, if the learned model $g(\Phi,\hat{\boldsymbol{\gamma}})$ is invariant across environments, the model $f_{\hat{\boldsymbol{\theta}}}$ treats all environments equally. As a result, a small perturbation on different environments should cause *the same model change*, and the variance of $\mathcal{IF}$ should be small. This leads to our proposal.

4.2 PROPOSED INDEX AND ITS UTILITY

We are now ready to address our main obstacle. On basis of environmental influence $\mathcal{IF}(\hat{\boldsymbol{\gamma}}, \mathbb{S}^e|\hat{\boldsymbol{\theta}})$ above, we propose our index to measure the fluctuation of the parameter change when different environments are upweighted:

$$\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}} := \ln\Big(\|\mathrm{Cov}_{e\in\mathcal{E}_{tr}}\big(\mathcal{IF}(\hat{\boldsymbol{\gamma}}, \mathbb{S}^e|\hat{\boldsymbol{\theta}})\big)\|_2\Big). \tag{5}$$

Here $\|\cdot\|_2$ is the 2-norm for matrix, $\mathrm{Cov}_{e\in\mathcal{E}_{tr}}(\cdot)$ refers to the covariance matrix of environmental influence across $\mathcal{E}_{tr}$ and $\ln(\cdot)$ is a nonlinear transformation that works well in practice. This measurement use 2-norm to represent the variation in $\mathcal{IF}(\hat{\boldsymbol{\gamma}}, \mathbb{S}^e|\hat{\boldsymbol{\theta}})$, and is practical as we will show.

OOD Model Generally, under most OOD problems, a good OOD model should (i) learn invariant and useful features; (ii) avoid spurious and varying features. Learning useful and invariant features means the model should have high test accuracy, no matter which test environment it is. In turn, high test accuracy also means the model truly learns some useful features. However, this is not enough, since we don't know whether the useful features are invariant features or spurious features. On the other hand, avoiding varying features means that different environments are *actually the same* to the model, so according to understanding in Section 4.1, $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ should be small. Combined this, we derive our proposal: *if a learnt model $f_{\hat{\boldsymbol{\theta}}}$ manage to achieve small $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ and high test accuracy simultaneously, it should have good OOD accuracy.* We prove our proposal in a simple but illuminating case, and we conduct various experiments (Section 5.2.2) to support our proposal. Several issues should be clarified. First, not all OOD problems demand models to learn invariant features, e.g. when varying features are always more strongly correlated to the prediction. But to our concern, we regard the OOD problem as a bridge to a truly causal model, so these minor "OOD" problems are out of our consideration. To a large extent, invariant features are still the major target and our proposal is still a good criterion to model's OOD property. Second, we admit that the gap between being stable in $\mathcal{E}_{tr}$ (small $\mathcal{V}$) and avoiding all spurious features on $\mathcal{E}_{all}$ does exist. However, to our knowledge, for features that are varying in $\mathcal{E}_{all}$ but are invariant in $\mathcal{E}_{tr}$, demanding a model to avoid them is somehow unrealistic. Therefore, we make a step forward that we measure whether the learnt model successfully avoids features that vary across $\mathcal{E}_{tr}$. We leave index about varying features in $\mathcal{E}_{all}$ to the future.

OOD Environment In addition to previous utility, the index can also discern whether P^e, $e \in \mathcal{E}_{tr}$ is different and whether an OOD algorithm is needed. Consider pooling all data points in $\mathbb{S}$ and randomly redistributed to m new environments $\{\tilde{\mathbb{S}}^1, \tilde{\mathbb{S}}^2, ..., \tilde{\mathbb{S}}^m\} := \tilde{\mathbb{S}}$. We implement a non-OOD algorithm (like ERM) on $\mathbb{S}$ to get $\hat{\boldsymbol{\theta}}$ and on $\tilde{\mathbb{S}}$ to get $\tilde{\boldsymbol{\theta}}$. Then, we compare $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ and $\mathcal{V}_{\tilde{\boldsymbol{\gamma}}|\tilde{\boldsymbol{\theta}}}$. If the former is much larger, it means $\boldsymbol{\theta}$ has learnt features that vary across $e \in \mathcal{E}_{tr}$, and so different environments are OOD under such algorithm. In this case, one should consider an OOD algorithm to achieve better generalization. Our understanding is that, if $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ is much larger than $\mathcal{V}_{\tilde{\boldsymbol{\gamma}}|\tilde{\boldsymbol{\theta}}}$, it means that using this algorithm, the model will exploit varying features, making the influence function in disparate environments quite different. Therefore, if we desire better OOD property by avoiding these features, we should implement an OOD algorithm. On the contrary, if $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ and $\mathcal{V}_{\tilde{\boldsymbol{\gamma}}|\tilde{\boldsymbol{\theta}}}$ are similar, we know that either the environments are not OOD or our available environments are too limited to learn an OOD model. In this case, we should either consider it as a non-OOD task or to collect more environments. Detailed experiments are in Section 5.2.1.

4.3 INFLUENCE CALCULATION

There is always a question surrounding the influence function: how can we efficiently calculate and inverse Hessian of a model's parameters? Koh & Liang (2017) suggests two methods (Conjugate Gradient and Stochastic estimation) to solve the problem. However, when $\hat{\boldsymbol{\theta}}$ is obtained by running SGD, it could hardly arrive at the global minimum. Although adding a damping term (i.e. let $\hat{\boldsymbol{H}}_{\hat{\boldsymbol{\theta}}} = \boldsymbol{H}_{\hat{\boldsymbol{\theta}}} + \lambda I$) can moderately alleviate the problem by transforming it into a convex situation, under large neural-network with non-linear activation function like ReLU, this method may still work poorly since the damping term in order to satisfy the transform is so large that it will influence the performance significantly. Most importantly, the variation of eigenvalue of Hessian is huge, making

the convergence of influence function calculation quite slow and inaccurate (Basu et al. (2020)), especially when the number of parameters is huge. The above problem limits the applicability of influence function in large neural networks.

In our proposed index, we circumvent the problem by excluding most parameters $\boldsymbol{\beta}$ and directly calculate Hessian of parameters $\boldsymbol{\gamma}$ to get accurate influence function value. This modification not only speed up the calculation, but it also coincides our expectation, that an OOD algorithm should learn invariant features *does not mean that* the influence function of *all* parameters should be identical across environments. For example, if $g(\Phi)$ wants to extract the same features from images in different environments, the influence function should be different on $\Phi(\cdot)$. Therefore, if we use all parameters to calculate the influence, given that $\boldsymbol{\gamma}$ is relatively insignificant in size compared with $\boldsymbol{\beta}$, the information of learnt features provided by $\boldsymbol{\gamma}$ is hard to be captured. On the contrary, considering influence function of only the top model g will manifest the influence of different environments in the aspect of *features*, thus enabling us to achieve our goal.

As our experiments show, after this modification, the influence function calculation speed can be 2000 times faster, and the utility (correlation with OOD property) could be even higher. One may not feel surprised given the huge number of parameters in embedding model $\Phi(\cdot)$. They slow down the calculation and overshadow top model's influence value.

5 EXPERIMENT

In this section, we experimentally show that: (1) A learnt model $f_{\boldsymbol{\theta}}$ will reach small $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ if it has good OOD property, while a non-OOD model won't. (2) The proposed index $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ provides additional information on the stability of the model, which overcome the weakness of the test accuracy. (3) The index can also check the difference between training environments to determine whether an OOD algorithm is needed.

We consider experiments in Bayesian Network and Colored MNIST respectively. The synthetic data generated by Bayesian Network includes environment-dependent noise and fake associations between features and response. For Colored MNIST, we know that the digit is the causal feature and the color is non-causal. This known causal relationships help us to determine the "worst" environment and obtain exact OOD accuracy. Note that, the problem here is to properly measure the *OOD generalization* rather than tuning for the SOTA performance on a test environment. There are other datasets widely used in domain generalization and domain adaptation, e.g. PACS, VLCS and so on. Considering the finite access to environments and the complicated causality behind images, we do not take these datasets into consideration.

5.1 SIMPLE BAYESIAN NETWORK

In this section, we show that the model with better OOD accuracy achieves smaller $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$. We assume the data is generated from the following Bayesian network:

$$\mathbf{x}_1 \leftarrow \mathcal{N}(0, \sigma_e^2), \quad \mathbf{y} \leftarrow \mathbf{x}_1^e \boldsymbol{W}_{1\to\mathbf{y}} + \mathcal{N}(0,1), \quad \mathbf{x}_2 \leftarrow \mathbf{y}^e \boldsymbol{W}_{\mathbf{y}\to 2} + \mathcal{N}(0, \sigma_e^2). \tag{6}$$

where $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^5$ are the features, $\mathbf{y} \in \mathbb{R}^5$ is the target vector, $\boldsymbol{W}_{1\to\mathbf{y}} \in \mathbb{R}^{5\times 5}$ and $\boldsymbol{W}_{\mathbf{y}\to 2} \in \mathbb{R}^{5\times 5}$ are the underlying parameters that are invariant across environments. The variance of gaussian noise is σ_e^2 that depends on environment. For simplicity, we denote $e = \sigma_e$ to represent an environment. The goal here is to linearly regress the response y on the input vector $(\boldsymbol{x}_1, \boldsymbol{x}_2)$, i.e. $\hat{\boldsymbol{y}} = \boldsymbol{x}_1 \hat{\boldsymbol{W}}_1 + \boldsymbol{x}_2 \hat{\boldsymbol{W}}_2$. According to the Bayesian network (6), $\mathbf{x}_1$ is the invariant (causal) feature, while the correlation between $\mathbf{x}_2$ and $\mathbf{y}$ is spurious and unstable since $e = \sigma_e$ varies across environments. Clearly, the model based only on $\mathbf{x}_1$ is an invariant model. Any invariant estimator should achieve $\hat{\boldsymbol{W}}_1 \approx \boldsymbol{W}_{1\to\mathbf{y}}$ and $\hat{\boldsymbol{W}}_2 \approx \mathbf{0}$.

Now consider five training environments $e \in \mathcal{E}_{tr} = \{0.2, 0.7, 1.2, 1.7, 2.2\}$, each containing 1000 data points. We estimate three linear models using ERM, IRM and REx respectively and record the parameter error as well as $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ (note that $\boldsymbol{\gamma}$ is $\boldsymbol{\theta}$ here). Table 1 presents the results among 500 repetitions. As expected, IRM and REx learn more invariant relationships than ERM (smaller causal error) and better avoid non-causal variables ($\hat{\boldsymbol{W}}_2 \approx \mathbf{0}$). Furthermore, the proposed measurement $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ is highly related to invariance, i.e. model with better OOD property achieves smaller $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$. This results coincides our understanding.

Table 1: Average parameter error $\|\hat{\boldsymbol{W}} - \boldsymbol{W}\|^2$ and the stable measurement $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ of 500 models from ERM, IRM and REx. Here, "Causal Error" represents $\|\hat{\boldsymbol{W}}_1 - \boldsymbol{W}_{1\to\mathbf{y}}\|^2$ and "Non-causal Error" represents $\|\hat{\boldsymbol{W}}_2\|^2$.

Method	$\mathcal{V}_{\boldsymbol{\gamma}\|\boldsymbol{\theta}}$	**Causal Error**	**Non-causal Error**
ERM	15.844	0.582	0.581
IRM	5.254	0.122	0.109
REx	1.341	0.042	0.033

5.2 Colored MNIST

Colored MNIST, derived from Arjovsky et al. (2019), introduces a synthetic binary classification task. The images are colored according to their binary label, making color a spurious feature in predicting the label. Specifically, for an environment e, we assign a preliminary binary label $\tilde{y} = \mathbf{1}_{\text{digits}\leq 4}$ and randomly flip $\tilde{y}$ with $p = 0.25$. Then, we color the image according to $\tilde{y}$ but with a flip rate of p^e. Clearly, when $p^e < 0.25$ or $p^e > 0.75$, color is more correlated with $\tilde{y}$ than real digit. Therefore, the oracle OOD model f_{OOD} will attain accuracy 0.75 in all environments while an ERM model may attain high training accuracy and low OOD property if p^e in training environments is too small or too large. Throughout the Colored MNIST experiments, we use three-layer MLP with ReLU activation and hidden dimension 256. Although our MLP model has relatively many parameters and is non-convex due to the activation layer, due to the technique mentioned in Section 4.3, the influence calculation is still fast and accurate, with directly calculating influence once spends less than 2 seconds.

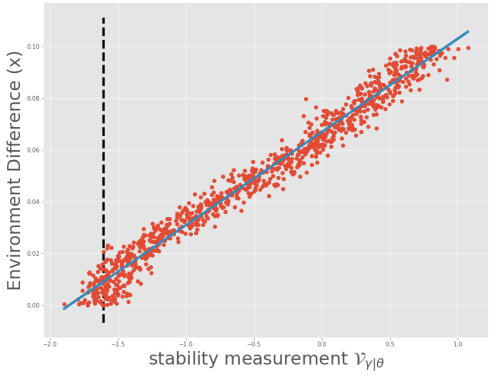

Figure 2: The index $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ is highly correlated to x. The plot contains 501 learnt ERM models with $x = 2 \times 10^{-4} i$, $i = 0, 1, ..., 500$. The dashed line is the baseline value when the difference between environments is eliminated by pooling and redistributing the training data. The blue solid line is the linear regression of x versus $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$. The Pearson coefficient is 0.9869, and the Spearman coefficient is 0.9873.

5.2.1 Identify OOD Problem

In this section, we show that $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ can discern whether our environments are OOD as mentioned in 4.2. Assume we have five training environments with

$$\mathcal{E}_{tr} = \big\{e : e \in \mathcal{E}_{all} \text{ and } p^e \in \{0.2 - 2x, 0.2 - x, 0.2, 0.2 + x, 0.2 + 2x\}\big\},$$

where $x \in [0.0, 0.1]$ represents the difference between training environments. If x is zero, all data points are generated from the same environment ($p^e = 0.2$) and so the task is not an OOD problem. On the contrary, larger x means out environments are more OOD. We repeat 501 times to learn the model with ERM. Given the learnt model $f_{\hat{\boldsymbol{\theta}}}$ and the training data, we compute $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ and check the correlation between $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ and x. Figure 2 presents the results. Our index $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ is highly related to x. The Pearson coefficient is 0.9869, and the Spearman coefficient is 0.9873. In addition, the benchmark of $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ that learns on the same training environments ($\tilde{\mathbb{S}}$ in 4.2) can be derived from the raw data by pooling and redistributing all data points, and we mark it by the black dashed line. If $\mathcal{V}_{\hat{\boldsymbol{\gamma}}|\hat{\boldsymbol{\theta}}}$ is much higher than the benchmark, indicating that x is not small, an OOD algorithm should be considered if better OOD generalization is demanded. Otherwise, present algorithm (like ERM) is sufficient. The results coincide our expectation that $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ can discern whether P^e is different.

5.2.2 RELATIONSHIP BETWEEN $\mathcal{V}$ AND OOD ACCURACY

In this section, we conduct experiment to support our proposal in Section 4.2. As previously proposed, if a model shows high test accuracy and small $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ simultaneously, it captures invariant features and avoids varying features, so it deserves to be an OOD model. In this experiment, we consider model with high test accuracy and show that smaller $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ generally corresponds to better OOD accuracy, which supports our proposal.

Consider two setups:

$$p^e \in \{0.0, 0.1\}$$

and

$$p^e \in \{0.1, 0.15, 0.2, 0.25, 0.3\}.$$

We implement IRM and REx with different penalty (note that ERM is $\lambda = 0$) to check relationship between $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ and OOD accuracy. For IRM and REx, we run 190 epochs pre-training without huge λ and use early stopping to prevent over-fitting. With these technique, all models (no matter ERM, IRM or REx) successfully achieve good test accuracy (within 0.1 of the oracle accuracy) and meet our requirement. Figure 3 presents the results. We can see that $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ are highly correlated to OOD accuracy in IRM and REx, with the absolute of Pearson Coefficient never less than 0.8417. Those models learned with larger λ present better OOD property, learning less varying features, and showing smaller $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$. The results are consistent with our proposal, except that when λ is large in IRM, $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ is a little bit unstable. We have carefully examined the phenomenon and found that it is caused by computational instability when inversing Hessian with eigenvalue quite close to 0. The problem of unstable inversing happens in limited algorithms with a low probability and can be addressed by repeating the experiment once or twice.

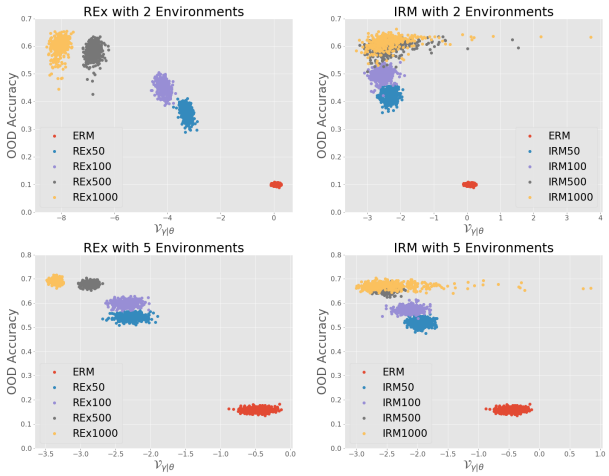

Figure 3: The relationship between $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ and OOD accuracy for models trained by REx (left) and IRM (right) with different $\lambda \in \{0, 50, 100, 500, 1000\}$. We train 400 models for each λ. The OOD accuracy and $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ enjoy high Pearson coefficient: -0.9745 (up-left), -0.9761 (down-left), -0.8417 (up-right), -0.9476 (down-right). The coefficients are negative because lower $\mathcal{V}_{\gamma|\boldsymbol{\theta}}$ forebodes better OOD property.

6 CONCLUSION

In this paper, we mainly focus on two presently unsolved problems, that how can we discern the OOD property of multiple environments and of learnt models. To this end, we introduce influence function into OOD problem and propose our index to help solve these issues. Our index can not only discern whether a multiple environments problem is OOD, but can also judge a model's OOD property when combined with test accuracy. To make our calculation more meaningful, accurate and efficient, we modify influence function to environmental level and propose to use only the top model to calculate the influence. Our method is proved in simple cases and it works well in experiments. We sincerely hope that, with the help of these index, our understanding of OOD generalization will become more and more precise and thorough.

REFERENCES

Kartik Ahuja, Karthikeyan Shanmugam, Kush Varshney, and Amit Dhurandhar. Invariant risk minimization games. *arXiv preprint arXiv:2002.04692*, 2020.

Kei Akuzawa, Yusuke Iwasawa, and Yutaka Matsuo. Domain generalization via invariant representation under domain-class dependency, 2019. URL https://openreview.net/forum?id=HJx38iC5KX.

Ahmed Alaa and Mihaela Van Der Schaar. Validating causal inference models via influence functions. volume 97 of *Proceedings of Machine Learning Research*, pp. 191–201, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

J Andrew Bagnell. Robust supervised learning. In *Proceedings of the national conference on artificial intelligence*, volume 20, pp. 714. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile. *arXiv preprint arXiv:2006.14651*, 2020.

Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 456–473, 2018.

Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust optimization*, volume 28. Princeton University Press, 2009.

Yoshua Bengio, Tristan Deleu, Nasim Rahaman, Nan Rosemary Ke, Sebastien Lachapelle, Olexa Bilaniuk, Anirudh Goyal, and Christopher Pal. A meta-transfer objective for learning to disentangle causal mechanisms. In *International Conference on Learning Representations*, 2019.

Peter Bühlmann et al. Invariance, causality and robustness. *Statistical Science*, 35(3):404–426, 2020.

Daniel C Castro, Ian Walker, and Ben Glocker. Causality matters in medical imaging. *Nature Communications*, 11(1):1–10, 2020.

Weiyu Cheng, Yanyan Shen, Linpeng Huang, and Yanmin Zhu. Incorporating interpretability into latent factor models via fast influence analysis. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 885–893, 2019.

Gilad Cohen, Guillermo Sapiro, and Raja Giryes. Detecting adversarial samples using influence functions and nearest neighbors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

R Dennis Cook. *Regression graphics: Ideas for studying regressions through graphics*, volume 482. John Wiley & Sons, 2009.

R Dennis Cook and Sanford Weisberg. Characterizations of an empirical influence function for detecting influential cases in regression. *Technometrics*, 22(4):495–508, 1980.

R Dennis Cook, Bing Li, et al. Dimension reduction for conditional mean in regression. *The Annals of Statistics*, 30(2):455–474, 2002.

Minghong Fang, Neil Zhenqiang Gong, and Jia Liu. Influence function based data poisoning attacks to top-n recommender systems. In *Proceedings of The Web Conference 2020*, pp. 3019–3025, 2020.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*, 2017.

Pang Wei W Koh, Kai-Siang Ang, Hubert Teo, and Percy S Liang. On the accuracy of influence functions for measuring group effects. In *Advances in Neural Information Processing Systems*, pp. 5254–5264, 2019.

Masanori Koyama and Shoichiro Yamaguchi. Out-of-distribution generalization with maximal invariant predictor. *arXiv preprint arXiv:2008.01883*, 2020.

David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). *arXiv preprint arXiv:2003.00688*, 2020.

Kun Kuang, Ruoxuan Xiong, Peng Cui, Susan Athey, and Bo Li. Stable prediction with model misspecification and agnostic distribution shift. In *AAAI*, pp. 4485–4492, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018.

Sara Magliacane, Thijs van Ommen, Tom Claassen, Stephan Bongers, Philip Versteeg, and Joris M Mooij. Domain adaptation by using causal inference to predict invariant conditional distributions. In *Advances in Neural Information Processing Systems*, pp. 10846–10856, 2018.

Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 5(78):947–1012, 2016.

James Robins, Lingling Li, Eric Tchetgen, Aad van der Vaart, et al. Higher order influence functions and minimax estimation of nonlinear functionals. In *Probability and statistics: essays in honor of David A. Freedman*, pp. 335–421. Institute of Mathematical Statistics, 2008.

James M Robins, Lingling Li, Rajarshi Mukherjee, Eric Tchetgen Tchetgen, Aad van der Vaart, et al. Minimax estimation of a functional on a structured high-dimensional model. *The Annals of Statistics*, 45(5):1951–1987, 2017.

Mateo Rojas-Carulla, Bernhard Schölkopf, Richard Turner, and Jonas Peters. Invariant models for causal transfer learning. *The Journal of Machine Learning Research*, 19(1):1309–1342, 2018.

Adarsh Subbaswamy, Peter Schulam, and Suchi Saria. Preventing failures due to dataset shift: Learning predictive models that transport. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 3118–3127. PMLR, 2019.

Daniel Ting and Eric Brochu. Optimal subsampling with influence functions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 3650–3659. Curran Associates, Inc., 2018. URL `http://papers.nips.cc/paper/7623-optimal-subsampling-with-influence-functions.pdf`.

Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pp. 1521–1528. IEEE, 2011.

Anastasios Tsiatis. *Semiparametric theory and missing data*. Springer Science & Business Media, 2007.

Mark J Van der Laan, MJ Laan, and James M Robins. *Unified methods for censored longitudinal data and causality*. Springer Science & Business Media, 2003.

Aad W Van der Vaart. *Asymptotic statistics*, volume 3. Cambridge university press, 2000.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pp. 831–838, 1992.

Sewall Wright. Correlation and causation. *J. agric. Res.*, 20:557–580, 1921.

Chuanlong Xie, Fei Chen, Yue Liu, and Zhenguo Li. Risk variance penalization: From distributional robustness to causality. *arXiv preprint arXiv:2006.07544*, 2020.

A APPENDIX

A.1 PROOF OF AN EXAMPLE

In this section, we use a simple model to illuminate the validity of $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}}$ proposed in Section 4. Consider a structural equation model (Wright (1921)):

$$\mathrm{x}_1 \sim P_x^e, \quad \mathrm{y} \leftarrow \mathrm{x}_1 + \mathcal{N}(0,1), \quad \mathrm{x}_2 \leftarrow \mathrm{y} + \mathcal{N}(0, \sigma_e^2)$$

where P_x^e is a distribution with a finite second-order moment, i.e. $\mathbb{E}\mathrm{x}_1^2 < +\infty$, and σ_e^2 is the variance of the noise term in x_2. Both P_x^e and σ_e^2 vary across environments. For simplicity, we assume there are infinite training data points collected from two training environments $\mathcal{E}_{tr} = \{(P_x^1, \sigma_1^2), (P_x^2, \sigma_2^2)\}$. Our goal is to predict y from $\mathbf{x} := (\mathrm{x}_1, \mathrm{x}_2)^\top$ using a least-squares predictor $\hat{y} = \boldsymbol{x}^\top \hat{\boldsymbol{\beta}} := x_1\hat{\beta}_1 + x_2\hat{\beta}_2$. Here we consider two algorithms: ERM and IRM with $\lambda \to +\infty$. According to Arjovsky et al. (2019), using IRM we obtain $\boldsymbol{\beta}_{\text{IRM}} \to (1,0)^\top$.

Intuitively, ERM will exploit both x_1 and x_2, thus achieving a better regression model. However, since relationship between y and x_2 varies across environments, our index will be huge in such condition. Conversely, β_{IRM} only uses invariant features x_1, thus $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}} \to -\infty$. Note that we do not have an embedding model here, so $\mathcal{V}_{\boldsymbol{\gamma}|\boldsymbol{\theta}} = \mathcal{V}_{\boldsymbol{\beta}}$.

ERM we denote

$$\ell(\boldsymbol{\beta}) = \frac{1}{|\mathcal{E}_{tr}|} \sum_{e \in \mathcal{E}_{tr}} \ell_e(\boldsymbol{\beta}) \quad \text{with} \quad \ell_e(\boldsymbol{\beta}) = \mathbb{E}_e(\mathrm{y} - \mathbf{x}\boldsymbol{\beta})^2.$$

Note that in $\mathbb{E}_e$, x_1 is sample from P_x^e. We then have

$$\frac{\partial \ell(\boldsymbol{\beta})}{\boldsymbol{\beta}} = -\sum_{e \in \mathcal{E}_{tr}} \mathbb{E}_e[\mathbf{x}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] = -\frac{2}{|\mathcal{E}_{tr}|} \sum_{e \in \mathcal{E}_{tr}} \begin{pmatrix} \mathbb{E}_e[\mathrm{x}_1(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] \\ \mathbb{E}_e[\mathrm{x}_2(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] \end{pmatrix}$$

To proceed further, we denote

$$\bar{d} = \frac{1}{|\mathcal{E}_{tr}|} \sum_{e \in \mathcal{E}_{tr}} \mathbb{E}_e \mathbf{x}_1^2, \quad s = \sum_{e \in \mathcal{E}_{tr}} \sigma_e^2 = \sigma_1^2 + \sigma_2^2.$$

By solving the following equations:

$$\frac{1}{|\mathcal{E}_{tr}|} \sum_{e \in \mathcal{E}_{tr}} \mathbb{E}_e[\mathrm{x}_1(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] = \bar{d}(1 - \beta_1 - \beta_2) = 0$$

and

$$\frac{1}{|\mathcal{E}_{tr}|} \sum_{e \in \mathcal{E}_{tr}} \mathbb{E}_e[\mathrm{x}_2(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] = (\bar{d} + 1)(1 - \beta_1 - \beta_2) + \beta_1 - \frac{s}{|\mathcal{E}_{tr}|}\beta_2 = 0$$

we have $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2)^\top$ with

$$\hat{\beta}_1 = \frac{s}{s+2}, \quad \hat{\beta}_2 = \frac{2}{s+2}.$$

Now we calculate our index. It is easy to see that

$$\begin{aligned} \frac{\partial \ell_e(\boldsymbol{\beta})}{\beta_1} &= -2\mathbb{E}_e[\mathrm{x}_1(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] = -2\mathbb{E}_e\mathrm{x}_1^2(1 - \beta_1 - \beta_2) \\ \frac{\partial \ell_e(\boldsymbol{\beta})}{\beta_2} &= -2\mathbb{E}_e[\mathrm{x}_2(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] = -2[(\mathbb{E}_e x_1^2 + 1)(1 - \beta_1 - \beta_2) + \beta_1 - \sigma_e^2\beta_2]. \end{aligned}$$

Therefore,

$$\nabla\ell_1(\boldsymbol{\beta}) - \nabla\ell_2(\boldsymbol{\beta}) = \begin{pmatrix} 0 \\ 2\beta_2(\sigma_1^2 - \sigma_2^2) \end{pmatrix} \quad \text{and} \quad \nabla\ell_1(\hat{\boldsymbol{\beta}}) - \nabla\ell_1(\hat{\boldsymbol{\beta}}) = \begin{pmatrix} 0 \\ \frac{4(\sigma_1^2 - \sigma_2^2)}{s+2} \end{pmatrix} \tag{7}$$

On the other hand, calculate the hessian and we have

$$\boldsymbol{H}_{\text{ERM}} = \begin{pmatrix} 2\bar{d} & 2\bar{d} \\ 2\bar{d} & 2\bar{d}+s+2 \end{pmatrix} \quad \text{and} \quad \boldsymbol{H}^{-1} = \frac{1}{2\bar{d}(s+2)} \begin{pmatrix} 2\bar{d}+s+2 & -2\bar{d} \\ -2\bar{d} & 2\bar{d} \end{pmatrix}.$$

Then we have (note that $\mathcal{IF}(\hat{\boldsymbol{\beta}}, \mathbb{S}^e) = \boldsymbol{H}^{-1}\nabla\ell_e(\hat{\boldsymbol{\beta}})$)

$$\begin{aligned}
\mathcal{V}_{\hat{\boldsymbol{\beta}}} &= \ln(\|\text{Cov}_{e\in\mathcal{E}}(\mathcal{IF}(\hat{\boldsymbol{\beta}}, \mathbb{S}^e))\|_2) \\
&= \ln(\frac{1}{4}\|(\mathcal{IF}_1 - \mathcal{IF}_2)(\mathcal{IF}_1 - \mathcal{IF}_2)^\top\|_2) \\
&= \ln(\frac{1}{4}\|\mathcal{IF}_1 - \mathcal{IF}_2\|^2) \\
&= 2\ln(\frac{1}{2}\|\boldsymbol{H}^{-1}(\nabla\ell_1(\hat{\boldsymbol{\beta}}) - \nabla\ell_2(\hat{\boldsymbol{\beta}}))\|) \\
&= 2\ln(\frac{1}{4\bar{d}(s+2)}\| \begin{pmatrix} 2\bar{d}+s+2 & -2\bar{d} \\ -2\bar{d} & 2\bar{d} \end{pmatrix} \begin{pmatrix} 0 \\ \frac{4(\sigma_1^2-\sigma_2^2)}{s+2} \end{pmatrix} \|) \\
&= 2\ln(\frac{2\sqrt{2}|\sigma_1^2 - \sigma_2^2|}{(s+2)^2})
\end{aligned}$$

where the third equation holds because the rank of matrix is 1. Clearly, when $|\sigma_1^2 - \sigma_2^2| \to 0$ (means two environments become identical), our index $\mathcal{V}_{\boldsymbol{\beta}} \to -\infty$. Otherwise, given $\sigma_1 \neq \sigma_2$, we have $\mathcal{V}_{\boldsymbol{\beta}} > -\infty$, showing that ERM captures varied features.

IRM We now turn to IRM model and show that $\mathcal{V}_{\boldsymbol{\beta}} \to -\infty$ when $\lambda \to +\infty$, thus proving IRM learnt model $\hat{\boldsymbol{\beta}}_{\text{IRM}}$ does achieve smaller $\mathcal{V}_{\boldsymbol{\beta}}$ compared with $\hat{\boldsymbol{\beta}}$ in ERM.

Under IRM model, assuming the tuning parameter is λ, we have

$$\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{|\mathcal{E}_{tr}|}\sum_{e\in\mathcal{E}_{tr}} \mathbb{E}_e[(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})^2] + 4\lambda\|\mathbb{E}_e[\mathbf{x}^\top\boldsymbol{\beta}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})]\|^2.$$

Then we have the gradient with respect $\boldsymbol{\beta}$:

$$\nabla\mathcal{L}(\boldsymbol{\beta}) = \frac{1}{|\mathcal{E}_{tr}|}\sum_{e\in\mathcal{E}_{tr}} \big(-2\mathbb{E}_e[\mathbf{x}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})] + 8\lambda\mathbb{E}_e[\mathbf{x}^\top\boldsymbol{\beta}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})]\mathbb{E}_e[\mathbf{x}(\mathrm{y} - 2\mathbf{x}^\top\boldsymbol{\beta})]\big),$$

and the Hessian matrix

$$\boldsymbol{H} = \boldsymbol{H}_{\text{ERM}} + \frac{8\lambda}{|\mathcal{E}|}\sum_{e\in\mathcal{E}} \big(\mathbb{E}_e[\mathbf{x}(\mathrm{y} - 2\mathbf{x}^\top\boldsymbol{\beta})]\mathbb{E}_e[\mathbf{x}(\mathrm{y} - 2\mathbf{x}^\top\boldsymbol{\beta})]^\top - 2\mathbb{E}_e[\mathbf{x}^\top\boldsymbol{\beta}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})]\mathbb{E}_e[\mathbf{x}\mathbf{x}^\top]\big). \tag{8}$$

Denote $\boldsymbol{\beta}_\lambda$ the solution of IRM algorithm on $\mathcal{E}_{tr}$ when penalty is λ. From Arjovsky et al. (2019) we know $\boldsymbol{\beta}_\lambda \to \boldsymbol{\beta}_{\text{IRM}} := (1, 0)^\top$. To show $\lim_{\lambda\to+\infty}\mathcal{V}_{\boldsymbol{\beta}_\lambda} = -\infty$, we only need to show that

$$\lim_{\lambda\to+\infty} \boldsymbol{H}^{-1}(\nabla\ell_1(\boldsymbol{\beta}_\lambda) - \nabla\ell_2(\boldsymbol{\beta}_\lambda)) = \mathbf{0}$$

We prove this by showing that

$$\lim_{\lambda\to+\infty} \boldsymbol{H}^{-1}(\lambda) = \mathbf{0} \quad \text{and} \quad \lim_{\lambda\to+\infty} \nabla\ell_1(\boldsymbol{\beta}_\lambda) - \nabla\ell_2(\boldsymbol{\beta}_\lambda) = \mathbf{0} \tag{9}$$

simultaneously. We add (λ) after $\boldsymbol{H}^{-1}$ to show that $\boldsymbol{H}^{-1}$ is a continuous function of λ. Rewrite $\boldsymbol{H}$ in formula 8 as

$$\boldsymbol{H}(\lambda, \boldsymbol{\beta}_\lambda) = \boldsymbol{H}_{\text{ERM}} + \lambda F(\boldsymbol{\beta}_\lambda)$$

where

$$\begin{aligned}
F(\boldsymbol{\beta}) &= \frac{8}{|\mathcal{E}_{tr}|}\sum_{e\in\mathcal{E}_{tr}} \big(\mathbb{E}_e[\mathbf{x}(\mathrm{y} - 2\mathbf{x}^\top\boldsymbol{\beta})]\mathbb{E}_e[\mathbf{x}(\mathrm{y} - 2\mathbf{x}^\top\boldsymbol{\beta})]^\top - 2\mathbb{E}_e[\mathbf{x}^\top\boldsymbol{\beta}(\mathrm{y} - \mathbf{x}^\top\boldsymbol{\beta})]\mathbb{E}_e[\mathbf{x}\mathbf{x}^\top]\big) \\
\lim_{\boldsymbol{\beta}_\lambda\to\boldsymbol{\beta}_{\text{IRM}}} F(\boldsymbol{\beta}_\lambda) &= \frac{4}{|\mathcal{E}_{tr}|}\sum_{e\in\mathcal{E}_{tr}} \begin{pmatrix} -\mathbb{E}_e\mathrm{x}_1^2 \\ 1 - \mathbb{E}_e\mathrm{x}_1^2 \end{pmatrix} \begin{pmatrix} -\mathbb{E}_e\mathrm{x}_1^2 & 1 - \mathbb{E}_e\mathrm{x}_1^2 \end{pmatrix} = F(\boldsymbol{\beta}_{\text{IRM}}) \text{ exists.}
\end{aligned}$$

Obviously, $F(\boldsymbol{\beta}_{\text{IRM}})$ is positive definite. Therefore, we have

$$\begin{aligned}\lim_{\lambda\to+\infty}\boldsymbol{H}(\lambda,\boldsymbol{\beta}_\lambda)^{-1} &= \lim_{\lambda\to+\infty}\lim_{\boldsymbol{\beta}_\lambda\to\boldsymbol{\beta}_{\text{IRM}}}[\boldsymbol{H}_{\text{ERM}}+\lambda F(\boldsymbol{\beta}_\lambda)]^{-1}\\ &= \lim_{\lambda\to+\infty}[\boldsymbol{H}_{\text{ERM}}+\lambda F(\boldsymbol{\beta}_{\text{IRM}})]^{-1}\\ &= \mathbf{0}\end{aligned}$$

The first equation holds because $\lim_{\lambda\to+\infty}F(\boldsymbol{\beta}_\lambda)=F(\boldsymbol{\beta}_{\text{IRM}})$ has the limit and is not $\mathbf{0}$, and the last equation holds because the eigenvalue of $\boldsymbol{H}$ goes to $+\infty$ when $\lambda\to+\infty$.

Now consider $\nabla\ell_1(\boldsymbol{\beta}_\lambda)-\nabla\ell_2(\boldsymbol{\beta}_\lambda)$. According to formula 7, we have

$$\begin{aligned}\lim_{\lambda\to+\infty}\nabla\ell_1(\boldsymbol{\beta}_\lambda)-\nabla\ell_2(\boldsymbol{\beta}_\lambda) &= \lim_{\boldsymbol{\beta}_\lambda\to\boldsymbol{\beta}_{\text{IRM}}}\nabla\ell_1(\boldsymbol{\beta}_\lambda)-\nabla\ell_2(\boldsymbol{\beta}_\lambda)\\ &= \nabla\ell_1(\boldsymbol{\beta}_{\text{IRM}})-\nabla\ell_2(\boldsymbol{\beta}_{\text{IRM}})\\ &= \begin{pmatrix}0\\ 2\beta_2(\sigma_1^2-\sigma_2^2)\end{pmatrix}\\ &= \mathbf{0}\end{aligned}$$

Hence we finish proof of formula 9 and show that $\mathcal{V}_{\boldsymbol{\beta}}\to-\infty$ in IRM.

A.2 Formula (4)

This section shows the derivation of the expression (4). Recall that the training dataset $\mathbb{S}=\{\mathbb{S}^1,...,\mathbb{S}^m\}$ and the objective function

$$\mathcal{L}(f,\mathbb{S})=\ell(f,\mathbb{S})+\lambda R(f,\mathbb{S}),$$

where the second term on the right hand side is the regularization. As to ERM, the regularization term is zero. Now considering upweight an environment $\mathbb{S}^e$. The new objective function is

$$\mathcal{L}_+(\boldsymbol{\theta},\mathbb{S},\delta)=\mathcal{L}(\boldsymbol{\theta},\mathbb{S})+\delta\cdot\ell(\boldsymbol{\theta},\mathbb{S}^e)$$

Notice that when upweight an environment, we only upweight the empirical loss on the corresponding environment. Further, we denote $\hat{\boldsymbol{\theta}},\hat{\boldsymbol{\theta}}_+$ as the optimal solutions before and after upweighting an environment. It is easy to see that $\|\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}\|\to0$ when $\delta\to0$. Following the derivation in Koh & Liang (2017), according to the first-order Taylor expansion of $\nabla\mathcal{L}_+(\boldsymbol{\theta},\mathbb{S},\delta)$ with respect to $\boldsymbol{\theta}$ on $\hat{\boldsymbol{\theta}}$,

$$\begin{aligned}\mathbf{0} &= \nabla[\mathcal{L}(\hat{\boldsymbol{\theta}}_+,\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}}_+,\mathbb{S}^e)]\\ &= \nabla(\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e))+\nabla^2[\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)](\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}})+o(\|\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}\|)\\ &= \delta\nabla\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)+\nabla^2[\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)](\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}})+o(\|\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}\|)\end{aligned}$$

Assume that $\nabla^2[\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)]$ is invertible, we have

$$\begin{aligned}\frac{\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}}{\delta} &= [\nabla^2\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})+\delta\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)]^{-1}\nabla\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)+o(\|\frac{\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}}{\delta}\|)\\ \lim_{\delta\to0}\frac{\hat{\boldsymbol{\theta}}_+-\hat{\boldsymbol{\theta}}}{\delta} &= [\nabla^2\mathcal{L}(\hat{\boldsymbol{\theta}},\mathbb{S})]^{-1}\nabla\ell(\hat{\boldsymbol{\theta}},\mathbb{S}^e)\end{aligned}$$

Note that this derivation is not fully rigorous. Please refer to Van der Vaart (2000) for more rigorous discussions about influence function.

A.3 Why not Accuracy

In Introduction, we have give an example where test accuracy mislead us. In this section, we will first supplement some examples where test accuracy not only misjudge different algorithms, but it also misjudge OOD property of models learnt with different penalty within the same algorithm. After that, we will show the universality of these problems and why test accuracy fails.

Consider two training environments

$$p^e\in\{0.0,0.1\},$$

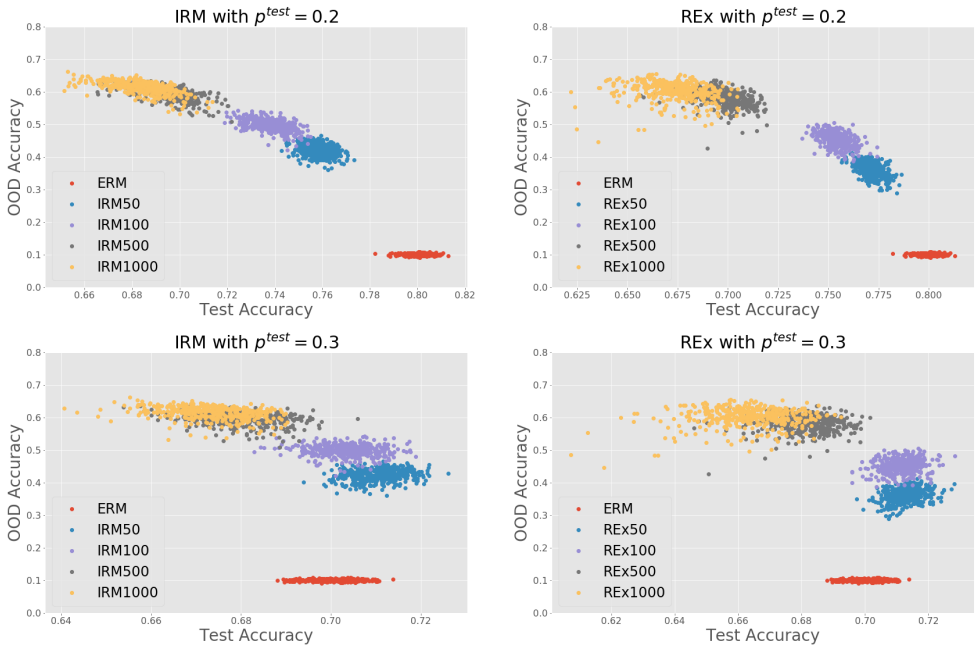

Figure 4: Experiments in Colored MNIST to show test accuracy (x-axis) cannot be used to judge model learnt with different penalty. Consider two test environments with $p^{\text{test}} = 0.2$ (up penals) and $p^{\text{test}} = 0.3$ (down penals). For each λ, we run IRM and REx 500 times. We can see that when λ increases from $\lambda = 0$ to $\lambda = 1000$, the OOD accuracy also increases, but test accuracy does not. When $p^{test} = 0.3$, their relationship becomes more perplexed.

and a test environment with flip rate denoted by p^{test}. We implement IRM and REx with penalty $\lambda \in \{0, 50, 100, 500, 1000\}$ to check the relationship between test accuracy and OOD accuracy. The training process is identical to experiment in section 5.2.2. As results showed in Figure 4, when OOD property of model gradually improves (caused by gradually increasing λ), it's relationship with test accuracy is either completely (when p^{test} is 0.2) or partly (when p^{test} is 0.3) negatively correlated. This phenomenon reveals the weakness of test accuracy. If one wants to select a λ when p^{test} is 0.3, judged by test accuracy, $\lambda = 50$ may be the best choice, no matter in IRM or REx. However, model learnt with $\lambda = 50$ has OOD accuracy even *less than a random guess model*.

Whether test accuracy is positively, negatively correlated or irrelevant to model's OOD property mainly depends on the "distance" between test environment and the "worst" environment for the model. If test accuracy happens to be the lowest among all the environments, we directly have OOD accuracy equals to test accuracy. In practice, however, their distance may be huge, and this is precisely the difficulty of OOD generalization. For example, we are accessible to images of cows in grasslands, woods and forests, but cows in desert are rare. At this point, the "worst" environment is certainly far from what we can get. If we expect model to capture real feature of cows, the model should avoid any usage of background color. However, a model based on color will perform consistently well (better than any OOD model) no matter in grasslands, woods and forests since all of available environments are green background in general. In Colored MNIST, test accuracy fails in the same way.

Such situations are actually quite common. Generally, within environments we have, there may be some features that are strongly correlated to the prediction but are slightly varied across environments. These features are spurious, given that their relationship with prediction is significantly disparate in other environments to which we want to generalize. However, using these features in prediction will easily achieve high test accuracy. Consequently, it will be extremely risky to judge models merely by test accuracy.