ZooD: Exploiting Model Zoo for Out-of-Distribution Generalization

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent advances on large-scale pre-training have shown great potentials of leveraging a large set of Pre-Trained Models (PTMs) for improving Out-of-Distribution (OoD) generalization, for which the goal is to perform well on possible unseen domains after fine-tuning on multiple training domains. However, maximally exploiting a zoo of PTMs is challenging since fine-tuning all possible combinations of PTMs is computationally prohibitive while accurate selection of PTMs requires tackling the possible data distribution shift for OoD tasks. In this work, we propose ZooD, a paradigm for PTMs ranking and ensemble with feature selection. Our proposed metric ranks PTMs by quantifying inter-class discriminability and inter-domain stability of the task data features extracted by the PTMs in a leave-one-domain-out cross-validation manner. The top-K ranked models are then aggregated for the target OoD task. To avoid accumulating noise induced by model ensemble, we propose an efficient variational EM algorithm to select informative features. We evaluate our paradigm on a diverse model zoo consisting of 35 models for various OoD tasks and demonstrate: (i) model ranking is better correlated with fine-tuning ranking than previous methods and up to 9859x faster than brute-force fine-tuning; (ii) OoD generalization outperforms the state-of-the-art methods and accuracy on most challenging task DomainNet is improved from 46.5% to 50.6%.

1 Introduction

Training and test data being Independent and Identically Distributed (IID) is a primary assumption behind most machine learning systems. However, this assumption does not hold in many real-world scenarios as real-world is marred with continuous distribution shifts [39]. Machine learning models encounter serious performance degradation [10, 32, 34] in such Out-of-Distribution (OoD) scenarios. To alleviate the accuracy degradation caused by distribution shifts, numerous algorithms have been proposed [4, 1, 40, 44, 6, 41, 71, 31, 21, 47, 7]. Recently, Gulrajani and Lopez-Paz [29] have argued for the systematic comparisons of OoD algorithms and introduced a standard and rigorous test bed called DomainBed. Their experimental comparison has raised some doubts about the effectiveness of OoD algorithms since they often fail to outperform the simple empirical risk minimization.

On the other hand, recent works [33, 2, 89, 65] have shown the advantages of pre-training for improving OoD generalization, i.e., learning from multiple training domains and being well applied to an unseen domain. The availability of a large set of Pre-Trained Models (PTMs) provides a possibility for solving various OoD tasks. However, it is challenging to sufficiently exploit the power of a model zoo (a large set of PTMs). One naive approach could be fine-tuning all possible combinations of PTMs on the target dataset and choosing the best performing one. However, naive fine-tuning is a costly and inflexible method with the risk of over-fitting [91]. Fine-tuning may also require exhaustive hyper-parameters search. Besides, fine-tuning becomes computationally prohibitive for a model zoo consisting of several hundred models and a dataset containing a large number of examples, making it impossible to use at any practical scale.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

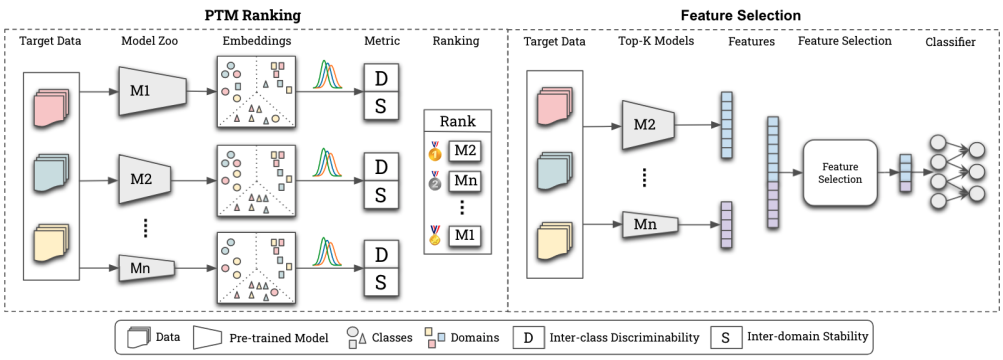

Figure 1: An overview of ZooD. Given a task with multiple training domains, the model ranking component evaluates and selects the top-K models that generalize well on this task. The features from selected models are then aggregated and denoised based on the feature selection component.

Recently, many ranking metrics have been proposed to estimate the transferability of models under IID assumption [9, 77, 59, 91, 90]. However, ranking a zoo of models for generalization on unseen distribution shifts is more challenging compared with IID setting. Moreover, even if a metric can correctly evaluate the transferability of each PTM, simply using the best model will not fully utilize rich knowledge present in a zoo of models. But the problem is even more serious that the most transferable model will include some noise, because noise and invariant features are undistinguishable in the sense that they are all stable across domains. Previous study [88] also pointed this out and emphasized the necessity of feature denoising. Therefore, if we leverage the model zoo by assembling relatively transferable models, the accumulation of noise features may increase memory use and hurt the predictive performance.

To solve the aforementioned problems, we propose ZooD, a paradigm to rank and aggregate a **Zoo** of PTMs for **OoD** generalization. An overview of our method is shown in Figure 1. Given a classification task with multiple training domains, to evaluate the generalization capability of each model, we quantify both the inter-class discriminability and inter-domain stability of the features extracted from each PTM in a leave-one-domain-out cross-validation manner, i.e., choosing one domain as the validation domain and each domain rotating as the validation domain, which is critical for identifying models that can extract domain-invariant features. Each PTM in the zoo is ranked by this quantification. ZooD then continues with model aggregation consisting of model ensemble and feature selection. By introducing latent masks over candidate features, an efficient EM algorithm is proposed to select informative features. To tackle the intractability of the posterior, variational approximation to the true posterior using a factorizable distribution is derived. We further extent it to large-scale datasets by building a local estimator under the stochastic approximation [66].

To demonstrate the efficacy of our method, we have performed extensive experiments with 35 diverse PTMs and 7 OoD datasets. First, we show that our ranking metric is strongly correlated with the fine-tuning performance of PTMs compared with existing IID metrics. Second, we illustrate the outstanding performance of ZooD on OoD datasets. For instance, on Office-Home, we get 85.1% average accuracy compared with previous SOTA of 70.6%. Lastly, we show the speedup of our method compared with brute-force fine-tuning. ZooD gives a maximum speedup of $\approx 10000\times$ (0.27 GPU hours vs 2662.27 GPU hours), making it practical and scalable.

Finally, to speed-up research and make our work more reproducible, we have devised a test bench consisting of extracted features, fine-tuning accuracy results, and ranking scores for all 35 PTMs in our model zoo. This testbed can help future research as the process of getting fine-tuning accuracy results based on DomainBed [29] for a zoo of models is computationally expensive. For instance, fine-tuning 35 models on all 7 OoD datasets costed approximately 35140 GPU hours (equivalent to 1464 GPU days or 4 GPU years). Concisely, our contributions are as follows:
(i) We propose an efficient and scalable ranking metric to gauge the generalization-ability of PTMs for unseen domains.
(ii) Using EM, we propose a method for selecting informative features and discarding invariant but noisy features in an ensemble of models.
(iii) We have established a test bed for PTMs on 7 OoD datasets, including features extracted by 35 PTMs in our model zoo, fine-tuning accuracy results and model ranking scores by different methods.

2 Related Work

Pre-training for OoD generalization. To tackle the problem of distribution shifts between training and test data, various OoD methods [4, 1, 40, 44, 23, 14, 6, 41, 71, 21, 47, 7] have been proposed with the aim to learn invariant representations across different environments. However, a standard evaluation [29] of many OoD algorithms shows that they do not significantly outperform simple ERM. On the other hands, recent works have shown effectiveness of pre-trained models for OoD generalization. Yi et al. [89] theoretically showed that adversarially pre-trained models also perform better for OoD generalization. Anonymous [3] performed a large-scale empirical analysis and show that the right choice of pre-trained models can achieve SOTA results. They also showed IID performance is not a good indicator of OoD performance and emphasized on the importance of model selection. Albuquerque et al. [2] showed the importance of feature extractor by proposing a new OoD-based pretext task for SSL pre-training that can outperform supervised training. CLIP [65] demonstrated that large-scale pre-training on a dataset of image-text pairs results in much more robust models for downstream tasks with various distribution shifts. Our work is based on these observations and we aim to facilitate utilization of PTMs by proposing an efficient metric as well as efficient feature ensemble and selection method.

Ranking pre-trained models by metric design. Large-scale, ever-increasing and evolving nature of PTMs requires a low-cost and flexible selection metric. Recently, a number of metrics have been introduced to estimate transferability of source-task-learned representations for target task under IID conditions. H-score [9] estimates the transferability by finding the relationship between extracted features and target class labels. NCE [77] proposes to estimate transferability via measuring conditional entropy between source and target labels. LEEP [59] simplifies NCE by using the joint distribution of source and target labels to estimate log expected empirical prediction. LogME [91, 90] estimates maximum value of label evidence given features from pre-trained models. The use of features instead of labels makes LogME more generalizable as it can be employed beyond classification. However, these transferability metrics focus on determining the compatibility of source-task-learned representations for the target task. We, on the other hand, aim to compute stability of these features across domains in addition to source-target transferability.

Ensemble and feature selection. Early works have shown that model ensemble can significantly improve predictive performance [22]. In the age of deep learning, Lakshminarayanan et al. [42] propose deep ensemble to measure predictive uncertainty. Similar works [61, 63] on uncertainty estimation focus on the context of outlier detection and reinforcement learning. When facing a zoo of PTMs, it's natural to leverage the rich knowledge by assembling multiple PTMs. In prior works, Liu et al. [49] propose using PTMs as teacher models that distill knowledge to a target model for downstream tasks. Shu et al. [72] propose Zoo-Tuning that learns to aggregate the parameters of multiple PTMs to a target model. However, these methods require the target model must have the identical architecture as the PTMs, thus sacrificing flexibility.

Our proposed paradigm involves selecting informative features from assembled feature extractors. In the related works of Bayesian variable selection, a prior is introduced over potential predictor subsets and subsequent method estimates posterior to identify promising subset models. Here we mainly focus on Stochastic search variable selection (SSVS) [60]. Meuwissen and Goddard [54] introduce a random effects variant of SSVS for gene mapping. Li and Zhang [45] consider regression modeling in high-dimensional spaces incorporating structural information. Ročková and George [67] propose EMVS for high-dimentional SSVS promising sparse high posterior probability submodels. Note that all aforementioned feature selection methods are only effective under the IID assumption, while in our paradigm, invariant and informative features can be selected from aggregated PTMs, which improves predictive performance for OoD tasks.

3 ZooD for OoD Generalization

3.1 Model Transferability Ranking

Assume that we have a domain distribution $\mathcal{D}$ from which we observe m domains: $\{\mathcal{D}_1,\mathcal{D}_2,\cdots,\mathcal{D}_m\}$. Each domain $\mathcal{D}_i$ is a set of (label, data) pairs, i.e. $\mathcal{D}_i = \{(y_{ij},x_{ij}),1 \leq j \leq n_i\}$. Meanwhile, we have a zoo of pre-trained feature extractors: $\mathcal{M} = \{\phi_1,\phi_2,\cdots,\phi_k,\cdots\}$. Our objective is to train a predictor f, along with one of the selected feature extractors from $\mathcal{M}$ (e.g., ϕ_k), such that the composed model $f \circ \phi_k$ performs well on both the m observed domains and unseen domains from $\mathcal{D}$.

In this work, we propose an algorithm that facilitates model selection *without* carrying out the fine-tuning step. For every model in $\mathcal{M}$, the algorithm produces an associated score, by which we can *rank* the models, such that the higher-ranked ones have a better chance to deliver stronger results after fine-tuning.

The proposed algorithm is a combination of 1) a model transferability metric and 2) a leave-one-domain-out cross-validation scheme. More specifically, we evaluate each feature extractor m times, and each time we treat the data from the held-out domain as validation data $\{(y'_j,x'_j)\}_{j=1}^{n'}$, while aggregating all remaining $(m-1)$ domains' data as the training data $\{(y_i,x_i)\}_{i=1}^{n}$. In the end, we average the m values of the model transferability metric. Finally, we rank all feature extractors in descending order of the average.

The transferability of each ϕ can be quantified in terms of *inter-class discriminability* and *inter-domain stability*. First, we denote the aggregated domain's label and feature as $\mathbf{y}=(y_1,...,y_n)^\top\in\mathbb{R}^n$ and $\Phi=\big(\phi(x_1),...,\phi(x_n)\big)^\top\in\mathbb{R}^{n\times d}$, respectively. We use $\mathbf{y}'\in\mathbb{R}^{n'}$ and $\Phi'\in\mathbb{R}^{n'\times d}$ for the held-out domain. Inter-domain stability is referring to correlation shift and covariate shift. Therefore, we formulate the objective as the following density function:

$$p(\mathbf{y}',\Phi'\big|\mathbf{y},\Phi)=p(\mathbf{y}'|\Phi',\mathbf{y},\Phi)p(\Phi'|\Phi),$$

where $p(\mathbf{y}'|\Phi',\mathbf{y},\Phi)$ measures discriminability and correlation shift between features Φ' and labels $\mathbf{y}'$, given the aggregated training data. Meanwhile, $p(\Phi'|\Phi)$ measures covariate shift between features Φ and Φ'. Given a hypothetical space $\mathcal{F}$ of classifiers, we can write $p(\mathbf{y}|\Phi)=\int_{f\in\mathcal{F}}p(\mathbf{y}|\Phi,f)p(f)\mathrm{d}f$. According to the Laplace approximation [51], if $p(\mathbf{y}|\Phi,f)$ is unimodal at $\boldsymbol{\mu}$, we can take Taylor expansion of the log-likelihood at the mode $\log p(\mathbf{y}|\Phi,f)\approx\log p(\boldsymbol{\mu}|\Phi,f)-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^\top\Lambda(\mathbf{y}-\boldsymbol{\mu})$, where $\Lambda=-\nabla_{\mathbf{y}^\top}\nabla_{\mathbf{y}}\log p(\mathbf{y}|\Phi,f)\big|_{\mathbf{y}=\boldsymbol{\mu}}$. The quadratic term implies that $p(\mathbf{y}|\Phi,f)$ can be approximated with a Gaussian distribution. Similar to You et al. [90], we consider a linear classifier, i.e. $f\circ\phi(\mathbf{x})=\mathbf{w}^\top\phi(\mathbf{x})$ with a Gaussian prior of $\mathbf{w}$:

$$\mathbf{w}\sim\mathcal{N}(\mathbf{0},\alpha^{-1}\mathbb{I}_d),\quad \mathbf{y}\big|\Phi,\mathbf{w}\sim\mathcal{N}(\Phi\mathbf{w},\beta^{-1}\mathbb{I}_n),$$

where α and β are two positive parameters. We estimate $\hat{\alpha}$ and $\hat{\beta}$ by maximizing the model evidence

$$p(\mathbf{y}|\Phi;\alpha,\beta)=\int_{\mathbf{w}\in\mathbb{R}^d}p(\mathbf{y}|\Phi,\mathbf{w};\beta)p(\mathbf{w};\alpha)\mathrm{d}\mathbf{w}$$

according to Algorithm 3 in You et al. [90] and compute the likelihood of $\mathbf{y}'$ as follows:

$$p(\mathbf{y}'|\Phi',\mathbf{y},\Phi;\hat{\alpha},\hat{\beta})=\frac{p(\mathbf{y}',\mathbf{y}|\Phi',\Phi;\hat{\alpha},\hat{\beta})}{p(\mathbf{y}|\Phi;\hat{\alpha},\hat{\beta})}.$$

For measuring covariate shift, we approximate the distribution of $\phi(x)$ with a Gaussian distribution $\mathcal{N}(\hat{\mu}_\phi,\hat{\Sigma}_\phi)$, where $\hat{\mu}_\phi$ and $\hat{\Sigma}_\phi$ are estimated from the training data Φ. Then we compute the density $p(\Phi'|\Phi)=p(\Phi'|\hat{\mu}_\phi,\hat{\Sigma}_\phi)$ to quantify the covariate shift.

Finally, we compute the density at the logarithmic scale and this defines the proposed metric

$$\text{Metric}=\log p(\mathbf{y}'|\Phi',\mathbf{y},\Phi)+\log p(\Phi'|\Phi). \tag{1}$$

Please refer to Appendix B.3 and B.4 for more details.

One distinctive aspect of our selection process is the cross-domain validation, embodied in the first term of (1). Across different domains, there are domain-invariant and domain-specific features, where over-fitting to the latter can severely harm the OoD generalization. By evaluating on held-out domains, we are able to filter out models that fixate on domain-specific features. To provide theoretical justification, an explicit analysis in the linear regression setting is conducted, where we show that the model with the optimal Metric is the one that select all domain-invariant features. Despite the over-simplification, it does reflect the essence of our approach. Due to page limit, the technical details are presented in Appendix B.5.

3.2 Model Ensemble with Feature Selection

The top-ranked PTMs in Section 3.1 are preferred for solving the OoD generalization task. To further aggregate different PTMs, we consider assembling the top-ranked feature extractors and rewrite $\Phi=\big[\Phi^{(1)},...,\Phi^{(k)}\big]$, where $\Phi^{(i)}$ is the feature matrix from the i-th ranked feature extractor.

As we show in experiments, in most cases, aggregating features from multiple models can significantly outperform any single model. However, simply concatenating features inevitably introduces more noise. As found in [88], non-informative but invariant features from training domains may only bring some noise, that is irrelevant to the classification problem, and the accumulation of noise hurts the learnability of the OoD generalization task while increasing the memory and computation cost. Therefore, we modify previous top linear model and present a feature selection tool under the Bayesian linear model framework in Section 3.1.

First, we impose a binary mask $\mathbf{z} = (\mathrm{z}_1, \mathrm{z}_2, ..., \mathrm{z}_d)^\top$ for the weight vector $\mathbf{w} = (\mathrm{w}_1, \mathrm{w}_2, ..., \mathrm{w}_d)^\top$, where $\mathrm{z}_i = 1$ indicates that w_i is an active weight in the top linear model, i.e. $\mathrm{w}_i \neq 0$, meaning the corresponding feature is informative, while $\mathrm{w}_i \approx 0$ if $\mathrm{z}_i = 0$, indicating a noisy feature that should be screened. Therefore the Bayesian feature selection is formulated by estimating the probability π_i of z_i with $\pi_i := p(\mathrm{z}_i = 1)$ and $\boldsymbol{\pi} = \{\pi_1, \pi_2, ..., \pi_d\}$.

To facilitate the utility of the mask, we assume that the weights $\{\mathrm{w}_i\}$ are independent of each other and each weight w_i is drawn from either a slab prior or a spike prior [37] with the mean of zero:

$$p(\mathrm{w}_i | \mathrm{z}_i, \alpha_{i,1}, \alpha_{i,2}) = \begin{cases} \mathcal{N}(0, \alpha_{i,1}^{-1}) & \text{if } \mathrm{z}_i = 1; \\ \mathcal{N}(0, \alpha_{i,2}^{-1}) & \text{if } \mathrm{z}_i = 0. \end{cases}$$

We make the Bayesian treatment to linear model in Section 3.1 by introducing gamma priors for all inverse variance terms:

$$\alpha_{i,1} \sim \text{Gamma}(\nu_{i,1}, \nu_{i,2}), \quad \alpha_{i,2} \sim \text{Gamma}(\nu_{i,3}, \nu_{i,4}), \quad \beta \sim \text{Gamma}(\nu_{0,1}, \nu_{0,2}),$$

and denote all hyper-parameters as $\boldsymbol{\nu} = \{\nu_{i,j}\}$. In addition, we denote all latent variables as $\boldsymbol{\xi} = \left\{\beta, \{\mathrm{w}_i, \mathrm{z}_i, \alpha_{i,1}, \alpha_{i,2}\}_{i=1}^d\right\}$. Under certain conditions, maximizing marginal likelihood provably leads to consistent selection and obeys Occam's razor phenomenon [27, 86], and thus screens non-informative features. To estimate π_i, the maximum marginal likelihood estimator of $(\boldsymbol{\pi}, \boldsymbol{\nu})$ is given by

$$\hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\nu}} = \underset{\boldsymbol{\pi}, \boldsymbol{\nu}}{\text{argmax}} \log p(\mathbf{y} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu}) = \underset{\boldsymbol{\pi}, \boldsymbol{\nu}}{\text{argmax}} \log \int_{\boldsymbol{\xi}} p(\mathbf{y}, \boldsymbol{\xi} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu}) \mathrm{d}\boldsymbol{\xi}.$$

However, direct maximization of (2) is intractable due to the integration over $\boldsymbol{\xi}$. EM algorithm might be a solution here [67]. In the E-step, we compute the conditional expectation:

$$\mathbb{E}_{\boldsymbol{\xi}} \left[\log p(\mathbf{y}, \boldsymbol{\xi} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu}) \middle| \mathbf{y}, \Phi; \boldsymbol{\pi}^{old}, \boldsymbol{\nu}^{old} \right].$$

Notice that evaluating the expectation involving the posterior distribution of $\boldsymbol{\xi}$. However in our case, it is not straightforward to obtain an analytical form of the true posterior distribution. We instead approximate it using Variational Inference [13] by introducing a tractable distribution Q. Considering the following objective function:

$$\mathcal{L}(Q) = \int_{\boldsymbol{\xi}} Q(\boldsymbol{\xi}; \boldsymbol{\pi}, \boldsymbol{\nu}) \log \frac{p(\mathbf{y}, \boldsymbol{\xi} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu})}{Q(\boldsymbol{\xi}; \boldsymbol{\pi}, \boldsymbol{\nu})} d\boldsymbol{\xi},$$

which is a lower bound of $\log p(\mathbf{y} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu})$. It has been shown the maximizer of $\mathcal{L}(Q)$ is the optimal approximator of $p(\boldsymbol{\xi} | \mathbf{y}, \Phi; \boldsymbol{\pi}, \boldsymbol{\nu})$ under the KL divergence. To obtain an explicit solution, we factorize Q into

$$Q(\boldsymbol{\xi}) = Q(\beta) \prod_{i=1}^{d} \Big[Q(\mathrm{z}_i) Q(\mathrm{w}_i) Q(\alpha_{i,1}) Q(\alpha_{i,2}) \Big], \tag{2}$$

which holds for the classical mean-field family. After all variational parameters in (2) are updated by running one-step coordinate gradient descent [13], in the M-step, we update $\boldsymbol{\pi}^{new}$ and $\boldsymbol{\nu}^{new}$ by maximizing:

$$\mathbb{E}_{\boldsymbol{\xi} \sim Q(\boldsymbol{\xi}; \boldsymbol{\pi}^{old}, \boldsymbol{\nu}^{old})} \left[\log p(\mathbf{y}, \boldsymbol{\xi} | \Phi; \boldsymbol{\pi}, \boldsymbol{\nu}) \right].$$

By repeating the E and M step, the estimator $(\boldsymbol{\pi}^{new}, \boldsymbol{\nu}^{new})$ converges to an optimal solution. We then screen those variables with converged prior π_i smaller than the predefined threshold τ. Our derivations for variational approximations and prior hyper-parameters optimization are listed in Appendix C.3.

However, the proposed algorithm still suffers from heavy computational cost: each iteration costs $\mathcal{O}(nd^2)$. To address this problem, we propose an efficient version based on Stochastic Variational Inference [35]. A local estimator $Q^s(\boldsymbol{\xi})$ is established under stochastic approximation that enjoys less computational complexity and guarantees convergence to global optimum [66]. We successfully reduce the computation cost to $\mathcal{O}(n^s d^2)$ with $n^s \ll n$. The complete algorithm is presented in Appendix C.4.

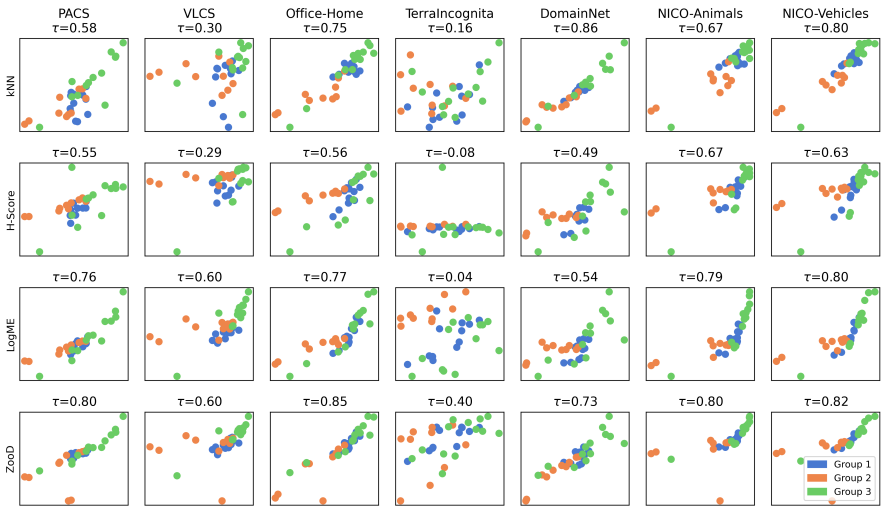

Figure 2: Comparison of ZooD ranking scores with three features-based ranking methods. The plots illustrate ground-truth out-of-domain accuracies (x-axis), ranking scores (y-axis) and Kendall's coefficient τ for 35 PTMs on seven datasets.

4 Experiments

In this section, we demonstrate the effectiveness of ZooD. First, we evaluate the ability of our ranking metric to estimate OoD performance and compare it with ground-truth performance and several existing IID ranking methods. Second, we show that our aggregation methods achieves significant improvements and SOTA results on several OoD datasets. Finally, we demonstrate that ZooD requires significantly less computation, and, therefore, is practically scalable compared with naive fine-tuning.

Setup Details. We use 35 PTMs with diverse architectures, pre-training methods and pre-training datasets. We divide the PTMs into three groups. Group 1 consists of models with different architectures, Group 2 consists of models pre-trained with different training methods, and Group 3 consists of models pre-trained on large-scale datasets. We conduct experiments on six OoD datasets: PACS [43], VLCS [25], Office-Home [78], TerraIncognita [11], DomainNet [64], and NICO (NICO-Animals & NICO-Vehicles) [31]. Each of the datasets has multiple domains. The standard way to conduct experiment is to choose one domain as test (unseen) domain and use the remaining domains as training domains, which is named leave-one-domain-out protocol. The top linear classifier is trained on the training domains only and tested on the test domain. Each domain rotates as the test domain and the average accuracy is reported for each dataset. To get ground-truth performance, we follow DomainBed [29] to fine-tune top linear classifiers for the PTMs on these OoD datasets. We adopt the leave-one-domain-out cross-validation setup in DomainBed with 10 experiments for hyper-parameter selection and run 3 trials. We triple the number of iterations for DomainNet (5000 to 15000) as it is a large-scale dataset requiring more iterations [18] and decrease the number of experiments for hyper-parameter selection from 10 to 5. More details on the experimental setup are in Appendix A.1.

4.1 Comparison with IID Ranking Metrics

IID ranking methods. We divide existing ranking methods into two groups. One group consists of methods that employ PTM's classification layer for ranking. These methods include NCE [77] and LEEP [59]. The other group consists of approaches that only use PTM's extracted features. These methods include H-Score [9] and LogME [91]. Additionally, we also use kNN with k=200 [82] as a baseline.

Evaluation metrics. To evaluate PTMs on OoD datasets with ranking methods, we follow leave-one-domain-out validation protocol [43]. For ZooD and kNN, we further adopt leave-one-domain-out validation for training domains and take average results as the performance prediction for the held-out test domain. To compute the correlation between ranking scores and ground-truth performance, we use two metrics. First, to compare the ranking of a transferability metric with accuracy, we employ Kendall's coefficient τ [38]. Unlike Pearson's correlation, τ measures correlation based on the order of

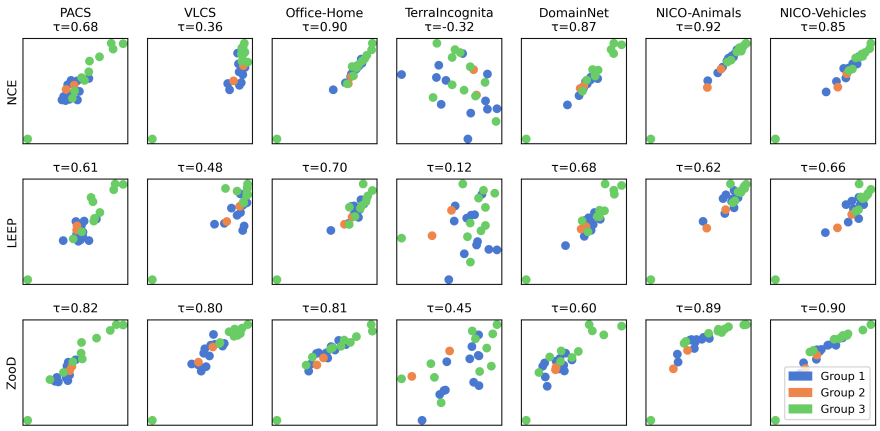

Figure 3: Comparison of ZooD ranking scores with two classification-layer based ranking methods. The plots illustrate ground-truth out-of-domain accuracies (x-axis), ranking scores (y-axis) and Kendall's coefficient τ for 25 PTMs that have classification layers on seven datasets.

Table 1: Comparisons: (a) τ_w between ZooD and feature-based transferability estimation methods using all of our PTMs. (b) τ_w between ZooD and classification-based transferability estimation methods. For this comparison, we consider 25 models that have classification heads. (c) Our method v.s. brute-force fine-tuning in terms of computing cost. For this comparison, we consider all 35 models.

(a) τ_w for feature based

	kNN	H-Score	LogME	ZooD
PACS	0.76	0.57	0.88	**0.91**
VLCS	0.49	0.45	0.79	**0.80**
Office-Home	0.78	0.68	**0.86**	**0.86**
TerraIncognita	0.40	-0.20	0.02	**0.46**
DomainNet	**0.89**	0.62	0.65	0.76
NICO-Animals	0.73	0.72	0.89	**0.90**
NICO-Vehicles	0.82	0.75	0.90	**0.92**

(b) τ_w for Classification based

	LEEP	NCE	ZooD
PACS	0.76	0.81	**0.89**
VLCS	0.57	0.32	**0.88**
Office-Home	0.76	**0.94**	0.86
TerraIncognita	0.02	-0.44	**0.59**
DomainNet	0.77	**0.87**	0.72
NICO-Animals	0.58	0.92	**0.94**
NICO-Vehicles	0.69	0.92	**0.95**

(c) Speed-up over brute-force

GPU Hours	ZooD	Fine-tuning	Speed Up
PACS	0.27	2662.27	9859×
VLCS	0.29	2706.67	9332×
Office-Home	0.39	3089.87	7922×
TerraIncognita	0.49	3920.27	8000×
DomainNet	11.24	17055.33	1516×
NICO-Animals	0.32	2914.40	9107×
NICO-Vehicles	0.30	2794.13	9313×

two measures. Consequently, it is a better criterion for ranking. Second, to measure the performance of transferability metric for top-model selection, we utilize weighted Kendall's coefficient τ_w [79]. The τ_w gives more weight to the ranking of top-performing models compared with the rest of the models. Therefore, it is a better comparative criterion for top model selection.

Results. First, we compare our method with feature-based scoring methods: kNN, H-Score, and LogME. These methods, similar to our method, rank models based on the penultimate layer. We compare ZooD with these methods for the full set of 35 PTMs. We plot ranking scores and ground-truth accuracies in Figure 2. For quantitative comparison, we also provide τ values. It can be seen that ZooD is better correlated with fine-tuning accuracy than other ranking methods on most of the datasets. For example, our method has a τ of 0.85 compared with LogME's τ of 0.77 on Office-Home and a τ of 0.40 compared with LogME's τ of 0.04 on TerraIncognita.

Furthermore, our metric is more stable and consistent. Precisely, τ of ZooD varies between 0.40 $\sim$ 0.85 compared with 0.04 $\sim$ 0.80 for LogME, -0.08 $\sim$ 0.67 for H-Score, and 0.16 $\sim$ 0.86 for kNN. The consistency of transferability metric across different datasets is critical since the purpose of a transferability metric is to estimate performance on a new dataset without having access to ground-truth accuracy. Whenever an estimation metric is inherently unstable, it is hard to determine its reliability for a new dataset.

Note that our method uses a linear model with Gaussian error to approximate the top classifier. This helps us achieve efficient model assessment, especially on small and medium-sized datasets in which the bias caused by model approximation is negligible compared with the estimation error due to insufficient data. However, on DomainNet, things may be different. The bias caused by model approximation dominants the evaluation performance on large datasets. Therefore, our method does not outperform kNN on DomainNet.

Second, we compare our method with classification-layer based methods: NCE and LEEP. For this comparison, we select a subset of our PTMs that have classification layers. The results are illustrated

Table 2: Comparison of out-of-domain accuracies between ZooD and SOTA OoD methods. The results of MixStyle [92] and SWAD [18] are from SWAD, and other results are from Gulrajani and Lopez-Paz [29] (denoted with †). Our results are average of three trials.

Method	PACS	VLCS	Office-Home	TerraInc.	Domain	Avg
ERM†	85.5	77.5	66.5	46.1	40.9	63.3
IRM†	83.5	78.6	64.3	47.6	33.9	61.6
GroupDRO†	84.4	76.7	66.0	43.2	33.3	60.7
I-Mixup†	84.6	77.4	68.1	47.9	39.2	63.4
MLDG†	84.9	77.2	66.8	47.8	41.2	63.6
MMD†	84.7	77.5	66.4	42.2	23.4	58.8
DANN†	83.7	78.6	65.9	46.7	38.3	62.6
CDANN†	82.6	77.5	65.7	45.8	38.3	62.0
MTL†	84.6	77.2	66.4	45.6	40.6	62.9
SagNet†	86.3	77.8	68.1	48.6	40.3	64.2
ARM†	85.1	77.6	64.8	45.5	35.5	61.7
VREx†	84.9	78.3	66.4	46.4	33.6	61.9
RSC†	85.2	77.1	65.5	46.6	38.9	62.7
MixStyle	85.2	77.9	60.4	44.0	34.0	60.3
SWAD	88.1	79.1	70.6	50.0	46.5	66.9
ZooD						
Single	96.0	79.5	84.6	37.3	48.2	69.1
Ensemble	95.5	80.1	85.0	38.2	50.5	69.9
F. Selection	**96.3**	**80.6**	**85.1**	42.3	**50.6**	**71.0**
F. Ratio (%)	24.3	24.5	62.5	76.8	99.8	

in Figure 3. It can be seen that ZooD is also more stable and consistent than NCE and LEEP. Moreover, Our method achieves superior performance on the difficult real-world TerraIncognita dataset. This dataset consists of obscure and blurry images captured by WildCams installed in different territories. NCE has a negative correlation for this dataset. On the other hand, our method, although not perfect, captures the relation in a better way. For this challenging dataset, our method has a τ of 0.45 compared with 0.12 and -0.32 for LEEP and NCE, respectively.

Third, we compare weighted Kendall's coefficient of our method with other ranking methods. The weighted Kendall's coefficient is a better metric to gauge the performance of a metric for top model selection. We also divide these results into two groups: comparison with feature-based scoring methods in Table 1a and comparison with classification-based scoring methods in Table 1b. Our method outperforms feature-based scoring methods on 6 out of 7 datasets. Similarly, it also outperforms both LEEP and NCE on 5 out of 7 datasets. Moreover, our ranking method is more stable as it performs better on challenging datasets. For example, it has τ_w of 0.46 $\sim$ 0.92 compared with LogME's τ_w of 0.02 $\sim$ 0.90 and H-Score's τ_w of -0.20 $\sim$ 0.75.

In summary, transferability estimation of ZooD correlates better with ground-truth accuracy on most of the OoD datasets compared with previous ranking methods. It also outperforms most feature-based metrics for model selection in terms of τ_w. Additionally, it is more stable and consistent across datasets, making it a better choice for pre-trained model selection.

4.2 SOTA Results with Our Selection Method

We also compare ZooD (model ranking and feature selection) with several recent SOTA OoD methods and demonstrate that it achieves substantial performance improvements. We compare previous OoD methods with three versions of our method: 1) **Single**: fine-tune the top-1 model by transferability metric; 2) **Ensemble**: fine-tune an ensemble of the top-K models; 3) **F. Selection**: fine-tune an ensemble of the top-K models with feature selection, which is the expected result using ZooD. By fine-tuning, we mean using ERM with DomainBed settings to fine-tune a top linear classifier for the PTMs. Their predictive performance and **F. Ratio** (the percentage of features used in **F. Selection**) are listed in the last four lines of Table 2.

In all experiment results, except TerraIncognita (discussed in the next paragraph), our method achieves remarkable improvement against ERM and recent SOTA. For **Single**, we list the improvements over

the previous SOTA as follows: +14% on Office-Home, +7.9% on PACS, +1.7% on DomainNet, and +0.4% on VLCS. This result also shows that even without aggregation, using proper pre-trained model can improve OoD generalization by a large margin.

The performance of **Single** does not outperform the previous SOTA on TerraIncognita. This is because previous methods fine-tune the whole network. In contrast, we only train a classifier on top of a fixed feature extractor. TerraIncognita is a much more challenging dataset compared with other OoD datasets, as the majority of its images are obscured by the background. Therefore it requires fully fine-tuning. To show the effectiveness of ZooD with fully fine-tuning, we select top-1 ranked model and fine-tune the whole model. Our resulted model achieves a +2.6% improvement compared with the previous SOTA. One limitation of ZooD when aggregating multiple models is that fine-tuning the whole models is difficult due to the limitation of GPU memory. However, for OoD tasks, fine-tuning the whole model may not perform better than fine-tuning the top classifier. For example, the results of fine-tuning the full top-ranked models on PACS, VLCS and Office-Home are 90.6, 79.1 and 83.4, respectively. Empirically, we find if a PTM is suitable for a given OoD task, fine-tuning the top classifier has better OoD generalization than fine-tuning the full model.

To efficiently utilize multiple models, we propose to select informative features in Section 3.2. Here, we compare the performance improvement by **F. Selection** with **Single** and **Ensemble**. ZooD significantly outperforms both candidates while only using a small portion of aggregated features from top-K models. Even on the most sophisticated DomainNet, ZooD can improve predictive performance by +2.4% compared with **Single** and +0.1% compared with **Ensemble**.

To find the appropriate number K for the model ensemble, we performed an ablation study. We varied the number of K, e.g. $K \in \{3, 5, 7\}$. The performance changes are plotted in Figure 4. We found the performance by aggregating top-3 models strikes the right balance between performance and computational complexity. Hence, $K = 3$ is set to the default value.

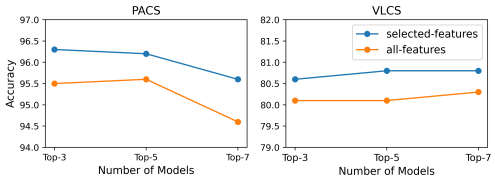

Figure 4: Comparison of selected-feature ensemble vs. all-feature ensemble for varying number of top models in the ensemble.

In summary, our ranking metric in ZooD is good enough to select a model that can outperform the previous SOTA methods without adding any bells and whistles. Furthermore, feature selection in ZooD can efficiently utilize informative features from top-K models to further improve the OoD generalization. Based on extensive experimental results on various OoD datasets, we conclude ZooD makes it easy and efficient to exploit a large set of PTMs for OoD generalization.

4.3 Computational Efficiency of ZooD

In the previous sections, we show its performance on several small and large-scale OoD datasets. Here, we illustrate the precision and computational efficiency of ZooD by comparing it with brute-force fine-tuning in terms of GPU hours. The results are shown in Table 1c. ZooD provides a minimum of 1516× speed-up for DomainNet and a maximum of 9859× speed-up for PACS. Cumulatively, our method took a total of 13 GPU hours to evaluate all the PTMs on all the datasets compared with 35140 GPU hours (equivalent to 4 GPU years) for brute-force fine-tuning. Therefore, ZooD is a scalable and practical method for OoD generalization.

5 Conclusion

Machine learning models rely on IID assumption, which is often violated due to constant distribution shifts in the real-world applications. In this work, we argue for leveraging a large set of PTMs to improve OoD generalization and propose ZooD, a paradigm for efficient PTMs ranking and aggregation. Our paradigm avoids the computationally-prohibitive fine-tuning by ranking PTMs based on quantifying their inter-class discriminability and inter-domain stability, and selecting the most informative features from top-ranked PTMs ensemble. Extensive experiments show ZooD is superior in ranking correlation with the ground-truth performance and achieves SOTA results on various OoD benchmarks.

References

[1] Kartik Ahuja, Karthikeyan Shanmugam, Kush Varshney, and Amit Dhurandhar. Invariant risk minimization games. In *International Conference on Machine Learning*, pages 145–155. PMLR, 2020.

[2] Isabela Albuquerque, Nikhil Naik, Junnan Li, Nitish Shirish Keskar, and Richard Socher. Improving out-of-distribution generalization via multi-task self-supervised pretraining. *ArXiv*, abs/2003.13525, 2020.

[3] Anonymous. An empirical study of pre-trained models on out-of-distribution generalization. In *Submitted to The Tenth International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=2RYOwBOFesi`. under review.

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

[5] Yuki M. Asano, C. Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *ArXiv*, abs/1911.05371, 2020.

[6] Haoyue Bai, Rui Sun, Lanqing Hong, Fengwei Zhou, Nanyang Ye, Han-Jia Ye, S-H Gary Chan, and Zhenguo Li. Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6705–6713, 2021.

[7] Haoyue Bai, Fengwei Zhou, Lanqing Hong, Nanyang Ye, S-H Gary Chan, and Zhenguo Li. Nas-ood: Neural architecture search for out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8320–8329, 2021.

[8] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[9] Yajie Bao, Yongni Li, Shao-Lun Huang, Lin Zhang, Lizhong Zheng, Amir Roshan Zamir, and Leonidas J. Guibas. An information-theoretic approach to transferability in task transfer learning. *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2309–2313, 2019.

[10] Andrei Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. Objectnet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Advances in Neural Information Processing Systems*, 32:9453–9463, 2019.

[11] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *ECCV*, 2018.

[12] Christopher M Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[13] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[14] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019.

[15] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[16] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Thirty-fourth Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

[17] George Casella, F Javier Girón, M Lina Martínez, and Elias Moreno. Consistency of bayesian procedures for variable selection. *The Annals of Statistics*, 37(3):1207–1228, 2009.

[18] Junbum Cha, Sanghyuk Chun, Kyungjae Lee, Han-Cheol Cho, Seunghyun Park, Yunsung Lee, and Sungrae Park. Swad: Domain generalization by seeking flat minima. *arXiv preprint arXiv:2102.08604*, 2021.

[19] Xinlei Chen, Haoqi Fan, Ross B. Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *ArXiv*, abs/2003.04297, 2020.

[20] Adrian Corduneanu and Christopher M Bishop. Variational bayesian model selection for mixture distributions. In *Artificial intelligence and Statistics*, volume 2001, pages 27–34. Morgan Kaufmann Waltham, MA, 2001.

[21] Joel Dapello, Tiago Marques, Martin Schrimpf, Franziska Geiger, David D Cox, and James J DiCarlo. Simulating a primary visual cortex at the front of cnns improves robustness to image perturbations. *BioRxiv*, 2020.

[22] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000.

[23] Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. *Advances in Neural Information Processing Systems*, 32: 6450–6461, 2019.

[24] Linus Ericsson, Henry Gouk, and Timothy M Hospedales. How well do self-supervised models transfer? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5414–5423, 2021.

[25] Chen Fang, Ye Xu, and Daniel N. Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. *2013 IEEE International Conference on Computer Vision*, pages 1657–1664, 2013.

[26] Edward I George and Robert E McCulloch. Approaches for bayesian variable selection. *Statistica sinica*, pages 339–373, 1997.

[27] Subhashis Ghosal, Jüri Lember, and Aad Van Der Vaart. Nonparametric bayesian model selection and averaging. *Electronic Journal of Statistics*, 2:63–89, 2008.

[28] Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.

[29] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021.

[30] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[31] Yue He, Zheyan Shen, and Peng Cui. Towards non-iid image classification: A dataset and baselines. *Pattern Recognition*, page 107383, 2020.

[32] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations*, 2019.

[33] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Xiaodong Song. Pretrained transformers improve out-of-distribution robustness. In *ACL*, 2020.

[34] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.

[35] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(5), 2013.

[36] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269, 2017.

[37] Hemant Ishwaran and J Sunil Rao. Spike and slab variable selection: frequentist and bayesian strategies. *The Annals of Statistics*, 33(2):730–773, 2005.

[38] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[39] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[40] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[41] Kun Kuang, Peng Cui, Susan Athey, Ruoxuan Xiong, and Bo Li. Stable prediction across unknown environments. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1617–1626, 2018.

[42] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30, 2017.

[43] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5543–5551, 2017.

[44] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[45] Fan Li and Nancy R Zhang. Bayesian variable selection in structured high-dimensional covariate spaces with applications in genomics. *Journal of the American statistical association*, 105(491):1202–1214, 2010.

[46] Junnan Li, Pan Zhou, Caiming Xiong, Richard Socher, and Steven C. H. Hoi. Prototypical contrastive learning of unsupervised representations. *ArXiv*, abs/2005.04966, 2021.

[47] Yichen Li and Xingchao Peng. Network architecture search for domain adaptation. *arXiv preprint arXiv:2008.05706*, 2020.

[48] Feng Liang, Rui Paulo, German Molina, Merlise A Clyde, and Jim O Berger. Mixtures of g priors for bayesian variable selection. *Journal of the American Statistical Association*, 103(481):410–423, 2008.

[49] Iou-Jen Liu, Jian Peng, and Alexander G Schwing. Knowledge flow: Improve upon your teachers. *arXiv preprint arXiv:1904.05878*, 2019.

[50] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *ArXiv*, abs/2103.14030, 2021.

[51] David J. C. MacKay. Choice of basis for laplace approximation. *Mach. Learn.*, 33(1):77–86, 1998.

[52] David JC MacKay. Probable networks and plausible predictions—a review of practical bayesian methods for supervised neural networks. *Network: computation in neural systems*, 6(3):469–505, 1995.

[53] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *ArXiv*, abs/1706.06083, 2018.

[54] Theo HE Meuwissen and Mike E Goddard. Mapping multiple qtl using linkage disequilibrium and linkage analysis information and multitrait data. *Genetics Selection Evolution*, 36(3):261–279, 2004.

[55] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6706–6716, 2020.

[56] Toby J Mitchell and John J Beauchamp. Bayesian variable selection in linear regression. *Journal of the american statistical association*, 83(404):1023–1032, 1988.

[57] Naveen Naidu Narisetty and Xuming He. Bayesian variable selection with shrinking and diffusing priors. *The Annals of Statistics*, 42(2):789–817, 2014.

[58] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[59] Cuong V Nguyen, Tal Hassner, C. Archambeau, and Matthias W. Seeger. Leep: A new measure to evaluate transferability of learned representations. In *ICML*, 2020.

[60] Robert B O'Hara and Mikko J Sillanpää. A review of bayesian variable selection methods: what, how and which. *Bayesian analysis*, 4(1):85–117, 2009.

[61] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29:4026–4034, 2016.

[62] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[63] Nick Pawlowski, Miguel Jaques, and Ben Glocker. Efficient variational bayesian neural network ensembles for outlier detection. *arXiv preprint arXiv:1703.06749*, 2017.

[64] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1406–1415, 2019.

[65] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021.

[66] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[67] Veronika Ročková and Edward I George. Emvs: The em approach to bayesian variable selection. *Journal of the American Statistical Association*, 109(506):828–846, 2014.

[68] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015.

[69] Hadi Salman, Andrew Ilyas, Logan Engstrom, Ashish Kapoor, and Aleksander Madry. Do adversarially robust imagenet models transfer better? *ArXiv*, abs/2007.08489, 2020.

[70] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018.

[71] Zheyan Shen, Peng Cui, Tong Zhang, and Kun Kunag. Stable learning via sample reweighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 5692–5699, 2020.

[72] Yang Shu, Zhi Kou, Zhangjie Cao, Jianmin Wang, and Mingsheng Long. Zoo-tuning: Adaptive transfer from a zoo of models. In *International Conference on Machine Learning*, pages 9626–9637. PMLR, 2021.

[73] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, D. Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[74] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, 2016.

[75] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[76] Bart Thomee, David A. Shamma, Gerald Friedland, Benjamin Elizalde, Karl S. Ni, Douglas N. Poland, Damian Borth, and Li-Jia Li. Yfcc100m: the new data in multimedia research. *Commun. ACM*, 59:64–73, 2016.

[77] A. Tran, Cuong V Nguyen, and Tal Hassner. Transferability and hardness of supervised classification tasks. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1395–1405, 2019.

[78] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5385–5394, 2017.

[79] Sebastiano Vigna. A weighted correlation index for rankings with ties. In *Proceedings of the 24th international conference on World Wide Web*, pages 1166–1176, 2015.

[80] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

[81] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020.

[82] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[83] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017.

[84] Xiaofan Xu and Malay Ghosh. Bayesian variable selection and estimation for group lasso. *Bayesian Analysis*, 10(4):909–936, 2015.

[85] Ismet Zeki Yalniz, Hervé Jégou, Kan Chen, Manohar Paluri, and Dhruv Kumar Mahajan. Billion-scale semi-supervised learning for image classification. *ArXiv*, abs/1905.00546, 2019.

[86] Yun Yang and Debdeep Pati. Bayesian model selection consistency and oracle inequality with intractable marginal likelihood. *arXiv preprint arXiv:1701.00311*, 2017.

[87] Yun Yang, Martin J Wainwright, and Michael I Jordan. On the computational complexity of high-dimensional bayesian variable selection. *The Annals of Statistics*, 44(6):2497–2532, 2016.

[88] Haotian Ye, Chuanlong Xie, Tianle Cai, Ruichen Li, Zhenguo Li, and Liwei Wang. Towards a theoretical framework of out-of-distribution generalization. *arXiv preprint arXiv:2106.04496*, 2021.

[89] Mingyang Yi, Lu Hou, Jiacheng Sun, Lifeng Shang, Xin Jiang, Qun Liu, and Zhi-Ming Ma. Improved ood generalization via adversarial training and pre-training. In *ICML*, 2021.

[90] Kaichao You, Yong Liu, Jianmin Wang, Michael I Jordan, and Mingsheng Long. Ranking and tuning pre-trained models: A new paradigm of exploiting model hubs. *arXiv preprint arXiv:2110.10545*, 2021.

[91] Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. Logme: Practical assessment of pre-trained models for transfer learning. In *International Conference on Machine Learning*, pages 12133–12143. PMLR, 2021.

[92] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. *ArXiv*, abs/2104.02008, 2021.

A Experiments

A.1 Complete Details of Experiment Setup

In this section, we provide a detailed experiment setup we have used. For completeness purposes, this section also includes details already mentioned in the main paper.

Pre-trained models. We use 35 PTMs having diverse architectures, pre-training methods and pre-training datasets. **Group 1** consists of models with different architectures. This group consists of 12 different architectures (CNNs and ViTs) trained on ImageNet-1k. The architectures are as follows: ResNet-50, ResNet-152 [30], ResNeXt-50 [83], DenseNet-169, DenseNet-201 [36], Inception v1 [73], Inception v3 [74], MobileNet v2 [70], EfficientNet-B2, EfficientNet-B4 [75], Swin-T, Swin-B [50]. **Group 2** consists of models pre-trained with different training methods. We use 10 ResNet-50s trained via following pre-training methods: Adversarial Training [53], BYOL [28], MoCo-v2 [19], InsDis [82], PIRL [55], DeepCluster-v2 [15], PCL-v2 [46], SeLa-v2 [5, 16], SwAV [16]. **Group 3** consists of models pre-trained on large-scale datasets. We used 13 different models trained on ImageNet-22k [68], YFCC-100M [76], IG-1B-Targeted [85], WebImageText [65]. A summary of the PTMs can be found in Table 3.

Datasets. We use six OoD datasets for our experiments. The details of these datasets are listed here. PACS [43] consists of 9,991 images from four domains (art, cartoons, photos, sketches) and seven classes. VLCS [25] consists of 10,729 images from four domains (Caltech101, LabelMe, SUN09, VOC2007) and five classes. Office-Home [78] has four domains (art, clipart, product, real) of common objects in office and home settings. The dataset has a total of 15,588 images belonging to 65 classes. TerraIncognita [11] contains photos of wild animals taken by camera traps installed at four different locations. It has a total of 24,788 images from 10 classes. DomainNet [64] is one of the most challenging OoD datasets. It has 586,575 images from six diverse domains (clipart, infographics, painting, quickdraw, real, sketch) belonging to 345 classes. NICO [31] consists of nearly 25,000 images from two superclasses: NICO-Animals (10 classes) and NICO-Vehicles (9 classes). We split the images of NICO-Animals and NICO-Vehicles into multiple domains according to [6] and combine validation and test sets as one domain to form four domains, separately.

Ground-truth performance. To get ground-truth performance, we train linear classifiers on top of PTMs following DomainBed [29]. The authors of DomainBed [29] argue for the hyper-parameter selection to be a part of the method selection criteria. Based on this argument, they propose a rigorous test bench. We follow their training and evaluation protocol, including dataset splits, hyper-parameter settings, optimizer, etc. We adopt the leave-one-domain-out cross-validation setup in DomainBed with 10 experiments for hyper-parameter selection and run 3 trials. We triple the number of iterations for DomainNet (5000 to 15000) as it is a larger dataset and requires more training [18] and decrease the number of experiments for hyper-parameter selection from 10 to 5.

IID ranking methods. We divide existing ranking methods into two groups. The first group consists of methods that employ PTM's classification layer for ranking. These methods include NCE [77] and LEEP [59]. The second group consists of approaches that only use PTM's extracted features. These methods include H-Score [9] and LogME [91]. Additionally, we also use kNN with k=200 [82] as a baseline.

Evaluation metrics. To evaluate PTMs on OoD datasets with ranking methods, we follow leave-one-domain-out validation protocol [43]. For ZooD and kNN, we further adopt leave-one-domain-out validation for training domains and take average results as the performance prediction for the held-out test domain. To compute the correlation between ranking scores and ground-truth performance, we use two metrics. First, to compare the ranking of a transferability metric with accuracy, we employ Kendall's coefficient τ [38]. Unlike Pearson's correlation, τ measures correlation based on the order of two measures. Consequently, it is a better criterion for ranking. Second, to measure the performance of transferability metric for top-model selection, we utilize weighted Kendall's coefficient τ_w [79]. The τ_w gives more weight to the ranking of top-performing models compared with the rest of the models. Therefore, it is a better comparative criterion for top model selection.

A.2 Extended Ranking Results

In this section, we provide detailed and raw results for all 35 models on all six OoD datasets. Specifically, we provide raw scores assigned by all the ranking methods to all PTMs. We also provide accuracy of

each model after fine-tuning. A more interpretable and visual analysis of these scores are provided in section 4.1 of the main paper.

We provide these raw scores here to help aid reproducibility and to help other researchers for easier benchmarking. It is important to note that getting these results, especially accuracy results, is computationally expensive, which may hinder future progress. For instance, on large DomainNet dataset, it takes 711 GPU days of training to get all ground-truth performance. Therefore, providing these raw scores can significantly help future researchers.

The results are provided in the following tables. Table 4 shows results on PACS and VLCS, Table 5 shows results on Office-Home and TerraIncognita, Table 6 contains results on NICO-Animals and NICO-Vehicles, and Table 7 contains results on DomainNet.

Table 3: Details of our model zoo. The first column corresponds to the numbers we have used for subsequent tables. The rest of the table describes architectures, pre-training datasets, and pre-training algorithms as well as the group and source of each model.

Number	Architecture	Dataset	Algorithm	Group	Source
1	ResNet-50	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
2	ResNet-152	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
3	ResNeXt-50	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
4	DenseNet-169	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
5	DenseNet-201	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
6	Inception v1	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
7	Inception v3	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
8	MobileNet v2	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
9	EfficientNet-B2	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
10	EfficientNet-B4	ImageNet-1K	ERM	Group 1	Paszke et al. [62]
11	Swin-T	ImageNet-1K	Swin	Group 1	Liu et al. [50]
12	Swin-B	ImageNet-1K	Swin	Group 1	Liu et al. [50]
13	ResNet-50	ImageNet-1K	Adv. ℓ_2 ($\epsilon=0.5$)	Group 2	Salman et al. [69]
14	ResNet-50	ImageNet-1K	Adv. ℓ_∞ ($\epsilon=4$)	Group 2	Salman et al. [69]
15	ResNet-50	ImageNet-1K	BYOL	Group 2	Ericsson et al. [24]
16	ResNet-50	ImageNet-1K	MoCo-v2	Group 2	Ericsson et al. [24]
17	ResNet-50	ImageNet-1K	InsDis	Group 2	Ericsson et al. [24]
18	ResNet-50	ImageNet-1K	PIRL	Group 2	Ericsson et al. [24]
19	ResNet-50	ImageNet-1K	DeepCluster-v2	Group 2	Ericsson et al. [24]
20	ResNet-50	ImageNet-1K	PCL-v2	Group 2	Ericsson et al. [24]
21	ResNet-50	ImageNet-1K	SeLa-v2	Group 2	Ericsson et al. [24]
22	ResNet-50	ImageNet-1K	SwAV	Group 2	Ericsson et al. [24]
23	ResNet-18	ImageNet-1K + YFCC-100M	Semi-supervised	Group 3	Yalniz et al. [85]
24	ResNet-50	ImageNet-1K + YFCC-100M	Semi-supervised	Group 3	Yalniz et al. [85]
25	ResNeXt-50	ImageNet-1K + YFCC-100M	Semi-supervised	Group 3	Yalniz et al. [85]
26	ResNeXt-101	ImageNet-1K + YFCC-100M	Semi-supervised	Group 3	Yalniz et al. [85]
27	ResNet-18	ImageNet-1K + IG-1B-Targeted	Semi-weakly Supervised	Group 3	Yalniz et al. [85]
28	ResNet-50	ImageNet-1K + IG-1B-Targeted	Semi-weakly Supervised	Group 3	Yalniz et al. [85]
29	ResNeXt-50	ImageNet-1K + IG-1B-Targeted	Semi-weakly Supervised	Group 3	Yalniz et al. [85]
30	ResNeXt-101	ImageNet-1K + IG-1B-Targeted	Semi-weakly Supervised	Group 3	Yalniz et al. [85]
31	Swin-B	ImageNet-1K + ImageNet-22K	Swin	Group 3	Liu et al. [50]
32	BEiT-B	ImageNet-1K + ImageNet-22K	BEiT	Group 3	Wolf et al. [80], Bao et al. [8]
33	ViT-B/16	ImageNet-1K + ImageNet-22K	ViT	Group 3	Wolf et al. [80], Wu et al. [81]
34	ResNet-50	WebImageText	CLIP	Group 3	Radford et al. [65]
35	ViT-B/16	WebImageText	CLIP	Group 3	Radford et al. [65]

Table 4: The ranking scores and fine-tuning accuracy on PACS and VLCS datasets. The numbering in the first column corresponds to a pre-trained model from Table 3. The numbers in each subsequent column represent the scores assigned by a ranking metric to the PTMs. The last column displays the accuracy of each model after fine-tuning. Empty cells represent models for which ranking is not feasible.

Model	PACS							VLCS						
Number	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.
1	-1.226	-1.077	5.016	49.608	0.226	0.053	66.9	-0.566	-0.498	3.241	58.156	0.223	0.119	76.7
2	-1.140	-1.007	5.072	54.767	0.274	0.100	74.4	-0.538	-0.494	3.253	61.215	0.229	0.127	77.0
3	-1.185	-1.022	5.010	50.737	0.231	0.064	65.6	-0.552	-0.499	3.216	58.540	0.200	0.083	76.9
4	-1.156	-0.998	4.636	43.284	0.186	-0.012	67.1	-0.569	-0.514	3.013	56.056	0.181	0.063	76.8
5	-1.172	-1.039	4.854	48.861	0.235	0.058	72.4	-0.581	-0.517	3.076	57.387	0.193	0.076	78.0
6	-1.392	-1.093	4.356	48.446	0.145	-0.025	65.3	-0.745	-0.549	2.811	58.260	0.136	0.004	74.6
7	-1.082	-0.947	4.795	37.655	0.164	-0.022	65.3	-0.565	-0.543	3.130	44.151	0.144	0.018	73.9
8	-1.209	-1.059	4.614	39.574	0.180	-0.002	65.0	-0.579	-0.512	2.922	59.465	0.152	0.030	75.9
9	-1.239	-0.949	4.857	46.069	0.270	0.067	74.2	-0.682	-0.505	3.002	58.049	0.131	-0.027	74.7
10	-0.993	-0.840	5.174	35.581	0.353	0.117	75.3	-0.556	-0.511	3.142	54.788	0.175	0.041	74.4
11	-1.231	-1.004	4.624	30.913	0.272	0.076	68.2	-0.637	-0.493	2.935	34.481	0.181	0.035	76.4
12	-1.154	-0.929	4.850	30.591	0.303	0.064	69.3	-0.601	-0.500	3.081	38.755	0.184	0.057	75.6
13	-1.230	-1.054	5.124	52.974	0.284	0.076	70.2	-0.584	-0.498	3.200	60.767	0.199	0.073	76.6
14	-1.226	-0.978	5.186	53.150	0.301	0.092	72.2	-0.667	-0.530	3.083	63.175	0.145	0.005	74.9
15			5.076	46.615	0.298	0.110	74.2			3.208	55.076	0.200	0.081	75.6
16			4.847	47.360	0.198	-0.075	58.9			3.260	60.138	0.247	0.141	69.8
17			4.578	31.131	0.066	-0.319	40.9			3.109	56.697	0.138	0.012	65.6
18			4.576	28.835	0.071	-0.309	38.4			3.150	55.033	0.162	0.043	64.2
19			5.024	36.493	0.256	-0.680	65.6			3.242	49.445	0.223	0.108	76.3
20			4.760	36.451	0.151	-0.093	58.4			3.205	54.922	0.209	0.102	71.3
21			4.829	35.495	0.187	-0.691	64.0			3.258	47.359	0.230	-0.435	75.4
22			4.946	34.103	0.231	0.034	62.9			3.253	52.114	0.231	0.119	77.1
23	-1.169	-0.974	4.225	48.668	0.190	0.034	69.4	-0.561	-0.503	2.832	57.624	0.214	0.107	77.1
24	-1.014	-0.908	5.181	57.411	0.362	0.164	75.7	-0.536	-0.503	3.340	58.396	0.313	0.208	78.6
25	-1.024	-0.881	5.151	55.490	0.312	0.099	74.4	-0.540	-0.500	3.312	62.857	0.268	0.173	77.8
26	-0.950	-0.841	5.287	61.007	0.369	0.156	78.4	-0.533	-0.505	3.340	63.100	0.285	0.190	77.9
27	-1.034	-0.834	4.609	63.988	0.302	0.159	83.4	-0.558	-0.484	2.828	58.549	0.211	0.105	77.0
28	-0.767	-0.630	5.499	75.592	0.578	0.400	91.7	-0.534	-0.495	3.363	61.016	0.341	0.238	79.1
29	-0.784	-0.612	5.493	78.550	0.531	0.358	89.0	-0.539	-0.493	3.347	62.604	0.302	0.203	78.1
30	-0.671	-0.518	5.625	74.917	0.646	0.447	91.5	-0.536	-0.499	3.371	66.276	0.312	0.211	78.7
31	-1.057	-0.740	5.587	41.936	0.527	0.263	85.4	-0.675	-0.499	3.163	39.618	0.275	0.176	78.6
32	-1.819	-1.415	3.424	26.731	-0.106	-0.214	47.1	-1.142	-0.794	2.048	52.277	-0.028	-0.213	68.4
33	-1.271	-0.995	4.621	58.167	0.198	-0.060	66.1	-0.601	-0.503	3.120	68.578	0.253	0.150	78.3
34			6.188	47.724	0.075	-0.106	66.0			3.198	64.808	0.275	0.184	74.9
35			5.546	84.858	0.869	0.653	96.0			3.143	67.367	0.377	0.312	79.5

B Model Ranking in ZooD

In this section, we present more details about the proposed ranking metric and algorithm.

B.1 Preliminaries: setup, problem and strategy

Suppose that:

- **Model zoo.** We have a collection of PTMs as learned feature extractors:

$$\mathcal{M}=\{\phi_1(x),\phi_2(x),...,\phi_k(x),...\},$$

  where $\phi_k(x)$ is a d-dimensional feature extractor that maps $\mathcal{X}$ to $\mathbb{R}^d$.

- **Dataset.** A multi-domain dateset is collected for solving a domain generalization problem:

$$\mathcal{D}=\{\mathcal{D}_1,\mathcal{D}_2,...,\mathcal{D}_m\}, \text{ with } \mathcal{D}_i=\big\{(x_{ij},y_{ij}),1\leq j\leq n_i\big\},$$

  where m is the number of observed domains and $\mathcal{D}_i$ is the set of data points under the i-th domain. The total sample size is $n=\sum_i n_i$.

- **Problem.** The objective is to select a PTM ϕ from $\mathcal{M}$ such that the optimal top classifier f based on the selected feature extractor ϕ, i.e. the whole predictor is $f\circ\phi(x)$, has good prediction performance on the domain generalization task.

To proceed further, we need more notations as folllows:

- For any domain i, we rewrite $\mathcal{D}_i=\{\mathbf{y}_i,\mathbf{x}_i\}$ where

$$\mathbf{y}_i=(y_{i1},y_{i2},...,y_{in_i})^\top\in\mathbb{R}^{n_i},\quad \mathbf{x}_i=(x_{i1},x_{i2},...,x_{in_i})^\top\in\mathbb{R}^{n_i\times p}.$$

Table 5: The ranking scores and fine-tuning accuracy for Office-Home and TeraIncognita datasets. The numbering in the first column corresponds to a pre-trained model from Table 3. The numbers in each subsequent column represent the scores assigned by a ranking metric to the PTMs. The last column displays the accuracy of each model after fine-tuning. Empty cells represent models for which ranking is not feasible.

Model	Office-Home							TerraIncognita						
Number	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.
1	-1.540	-1.311	41.908	50.614	0.985	0.075	67.7	-1.531	-1.286	5.559	23.477	0.301	-0.722	31.0
2	-1.355	-1.198	43.973	53.499	1.029	0.120	70.6	-1.501	-1.338	5.592	29.018	0.305	-0.721	35.2
3	-1.465	-1.263	41.439	51.501	0.979	0.076	69.1	-1.519	-1.290	5.491	23.227	0.292	-0.735	25.5
4	-1.457	-1.280	35.695	47.413	0.941	0.025	68.7	-1.473	-1.266	4.850	22.977	0.244	-0.815	23.9
5	-1.460	-1.271	37.727	48.186	0.952	0.036	69.1	-1.573	-1.321	5.119	22.116	0.251	-0.831	23.0
6	-2.243	-1.701	30.175	44.089	0.887	-0.015	59.0	-1.636	-1.327	4.432	24.368	0.238	-0.881	17.7
7	-1.396	-1.327	40.696	53.520	0.977	0.083	66.2	-1.440	-1.286	5.097	24.285	0.250	-0.819	23.8
8	-1.713	-1.439	32.911	45.934	0.902	-0.005	62.8	-1.614	-1.373	4.782	22.793	0.264	-0.811	29.7
9	-1.628	-1.143	40.378	51.252	1.022	0.106	72.2	-1.610	-1.388	5.124	25.737	0.299	-0.740	32.8
10	-1.229	-1.082	45.309	45.939	1.094	0.176	73.6	-1.523	-1.383	5.517	25.909	0.319	-0.720	24.8
11	-1.528	-1.174	36.781	47.708	1.018	0.100	72.5	-1.563	-1.393	4.474	26.624	0.272	-0.746	30.3
12	-1.320	-1.099	42.086	48.265	1.070	0.139	75.9	-1.545	-1.466	4.984	25.561	0.289	-0.720	30.9
13	-1.594	-1.311	41.423	48.194	0.972	0.061	66.3	-1.625	-1.315	6.101	25.319	0.348	-0.803	31.9
14	-1.825	-1.377	39.631	43.415	0.937	0.027	62.4	-1.704	-1.309	6.106	24.481	0.344	-0.910	26.7
15			40.498	37.124	0.971	-0.022	60.6			5.542	24.565	0.307	-0.721	23.7
16			38.633	32.130	0.941	-0.102	41.6			5.601	26.435	0.308	-0.742	19.1
17			31.841	18.154	0.825	-0.399	22.7			5.675	27.931	0.308	-1.067	16.0
18			32.493	19.447	0.838	-0.366	24.4			5.711	30.123	0.313	-0.777	18.4
19			39.876	30.521	0.956	-0.010	61.0			5.649	26.656	0.322	-0.710	28.7
20			36.612	27.949	0.912	-0.100	44.1			5.486	23.898	0.296	-0.775	16.1
21			38.936	29.547	0.950	-0.424	52.7			5.537	23.617	0.303	-0.745	23.6
22			39.705	28.988	0.954	-0.041	58.8			5.680	26.854	0.323	-0.994	23.2
23	-1.680	-1.400	26.787	45.371	0.895	-0.028	62.3	-1.560	-1.311	3.817	23.495	0.228	-0.846	26.5
24	-1.339	-1.194	44.073	49.205	1.049	0.097	71.2	-1.487	-1.322	5.527	25.801	0.309	-0.698	32.5
25	-1.294	-1.156	44.683	56.220	1.055	0.151	72.7	-1.505	-1.335	5.439	24.983	0.291	-0.718	27.7
26	-1.168	-1.081	46.671	60.344	1.106	0.199	74.8	-1.487	-1.360	5.510	26.461	0.302	-0.685	28.8
27	-1.502	-1.266	28.820	49.142	0.924	0.004	66.7	-1.549	-1.291	3.761	23.208	0.223	-0.856	29.3
28	-1.152	-1.024	46.552	56.192	1.119	0.167	76.1	-1.495	-1.354	5.428	25.008	0.298	-0.739	36.0
29	-1.111	-0.979	47.382	61.253	1.133	0.230	78.0	-1.515	-1.360	5.342	26.525	0.277	-0.730	34.4
30	-0.971	-0.875	50.223	67.685	1.226	0.312	81.0	-1.449	-1.343	5.478	28.274	0.298	-0.681	35.4
31	-1.252	-0.859	47.500	60.458	1.240	0.306	84.6	-1.579	-1.392	4.934	29.336	0.303	-0.669	37.3
32	-3.896	-2.913	15.908	9.459	0.755	-0.178	31.9	-1.828	-1.400	19.076	24.408	0.230	-0.939	26.2
33	-1.675	-1.295	37.045	58.928	1.027	0.107	71.8	-1.548	-1.268	-0.153	26.017	0.247	-0.827	21.3
34			26.080	22.301	0.828	-0.091	42.4			3.695	28.290	0.220	-0.868	18.8
35			36.712	65.789	1.056	0.148	82.2			4.147	31.467	0.259	-0.749	40.0

- Given a feature extractor ϕ, the learned feature matrix is denoted by

$$\Phi_i = \big(\phi(x_{i1}),\phi(x_{i2}),...,\phi(x_{in_i})\big)^\top \in \mathbb{R}^{n_i \times d}.$$

- For any $i \in [m]$, we denote Φ_{-i} and $\mathbf{y}_{-i}$ as

$$\begin{aligned} \mathbf{y}_{-i} &= \big(\mathbf{y}_1^\top,\cdots,\mathbf{y}_{i-1}^\top,\mathbf{y}_{i+1}^\top,\cdots,\mathbf{y}_m^\top\big)^\top \in \mathbb{R}^{(n-n_i)}, \\ \Phi_{-i} &= \big(\Phi_1^\top,\cdots,\Phi_{i-1}^\top,\Phi_{i+1}^\top,\cdots,\Phi_m^\top\big)^\top \in \mathbb{R}^{(n-n_i)\times d}. \end{aligned}$$

We can break the model selection problem down into two questions. 1). When generalizing to unknown domains, are the learned features stable enough to avoid extrapolating predictions? 2). Are the learned features informative enough to ensure that the correlation between features and labels is stable across different domains? To answer these two questions, we compute the following two quantities:

- $p(\Phi_i|\Phi_{-i})$, which measures covariate shift between Φ_i and Φ_{-i}, indicating whether the validation input is a rare sample compared with the training input;
- $p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i})$, which measures the discriminability and correlation shift between Φ_i and $\mathbf{y}_i$ given the training data Φ_{-i} and $\mathbf{y}_{-i}$.

We thus propose a metric by assembling above quantities for PTMs ranking:

$$\text{Metric} = \log p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i}) + \lambda \log p(\Phi_i|\Phi_{-i}), \tag{3}$$

where λ is a tuning parameter that unifies the scale of the correlation shift and the covariate shift. In our implementation, the tuning parameter is taken to be the ratio of two standard deviations:

$$\lambda = \frac{\text{Std}(\log p(y_{ij}|\Phi_i,\mathbf{y}_{-i},\Phi_{-i}))}{\text{Std}(\log p(\phi(x_{ij})|\Phi_{-i}))},$$

which is also used in Ye et al. [88].

Table 6: The ranking scores and fine-tuning accuracy for NICO dataset. The numbering in the first column corresponds to a pre-trained model from Table 3. The numbers in each subsequent column represent the scores assigned by a ranking metric to the PTMs. The last column displays the accuracy of each model after fine-tuning. Empty cells represent models for which ranking is not feasible.

Model	NICO-Animal							NICO-Vehicle						
Number	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.
1	-0.501	-0.397	7.767	86.348	0.512	0.510	91.0	-0.699	-0.651	6.758	81.043	0.398	0.363	86.1
2	-0.419	-0.340	7.975	88.823	0.599	0.602	92.8	-0.624	-0.598	6.928	84.266	0.467	0.433	88.1
3	-0.455	-0.379	7.789	87.400	0.527	0.525	92.0	-0.670	-0.637	6.773	82.466	0.405	0.376	86.7
4	-0.450	-0.376	7.358	86.748	0.466	0.455	91.8	-0.692	-0.661	6.387	80.092	0.370	0.329	86.4
5	-0.479	-0.375	7.472	85.773	0.483	0.471	92.0	-0.720	-0.679	6.469	79.118	0.381	0.340	86.6
6	-0.983	-0.629	6.721	78.237	0.343	0.326	83.7	-1.109	-0.834	5.718	72.119	0.242	0.206	79.2
7	-0.460	-0.450	7.748	84.286	0.519	0.502	88.7	-0.647	-0.660	6.659	77.803	0.371	0.336	83.6
8	-0.616	-0.508	6.810	81.108	0.326	0.318	86.6	-0.792	-0.743	5.959	76.653	0.268	0.233	82.5
9	-0.646	-0.345	7.814	79.292	0.600	0.583	92.1	-0.823	-0.600	6.739	80.821	0.474	0.427	88.0
10	-0.393	-0.318	8.089	82.033	0.693	0.664	92.4	-0.578	-0.560	7.016	77.957	0.547	0.493	88.0
11	-0.598	-0.309	7.797	80.542	0.681	0.656	93.5	-0.742	-0.569	6.655	80.318	0.526	0.477	89.1
12	-0.460	-0.277	8.201	80.414	0.811	0.798	95.1	-0.644	-0.545	6.963	78.615	0.593	0.545	90.3
13	-0.602	-0.468	7.551	82.090	0.433	0.428	88.0	-0.743	-0.651	6.685	78.180	0.374	0.340	84.9
14	-0.921	-0.634	7.030	69.756	0.288	0.275	81.2	-0.941	-0.731	6.407	71.239	0.283	0.247	80.7
15			7.546	71.552	0.438	0.427	86.9			6.644	71.200	0.362	0.326	82.9
16			7.679	73.400	0.491	0.485	80.0			6.701	67.634	0.376	0.331	74.0
17			6.562	46.842	0.188	0.166	53.2			6.050	49.714	0.184	0.143	53.6
18			6.756	48.977	0.225	0.207	55.4			6.184	52.048	0.221	0.176	56.0
19			7.652	68.655	0.470	0.462	89.3			6.743	69.965	0.395	0.354	83.8
20			7.491	68.446	0.429	0.419	81.1			6.532	65.629	0.323	0.276	75.6
21			7.649	60.005	0.458	-0.970	84.2			6.681	62.967	0.370	-0.704	78.3
22			7.580	65.025	0.445	0.436	87.7			6.710	66.776	0.385	0.343	82.4
23	-0.482	-0.391	6.713	84.406	0.404	0.391	90.6	-0.688	-0.633	5.748	77.967	0.324	0.284	85.9
24	-0.346	-0.278	8.081	89.122	0.666	0.656	94.3	-0.593	-0.573	7.001	83.783	0.524	0.479	89.9
25	-0.333	-0.255	8.266	88.655	0.754	0.757	95.1	-0.563	-0.538	7.122	86.015	0.559	0.519	90.1
26	-0.305	-0.245	8.383	89.750	0.832	0.831	95.9	-0.524	-0.514	7.250	87.605	0.627	0.582	91.1
27	-0.444	-0.347	6.793	81.971	0.425	0.410	91.3	-0.649	-0.602	5.873	78.824	0.350	0.312	86.4
28	-0.283	-0.211	8.253	89.394	0.772	0.762	95.8	-0.527	-0.520	7.131	85.509	0.594	0.549	91.1
29	-0.287	-0.192	8.424	93.119	0.872	0.871	96.7	-0.515	-0.490	7.250	88.538	0.632	0.590	91.6
30	-0.255	-0.164	8.594	90.335	1.038	1.037	97.4	-0.478	-0.450	7.430	89.605	0.752	0.710	92.8
31	-0.521	-0.167	8.407	84.414	1.086	1.063	97.5	-0.641	-0.439	7.254	90.010	0.824	0.774	94.5
32	-1.864	-1.317	4.772	35.264	0.057	0.031	62.2	-1.801	-1.282	4.525	41.243	0.044	0.007	64.4
33	-0.393	-0.224	8.673	93.392	0.819	0.798	94.6	-0.616	-0.511	6.808	89.564	0.589	0.534	90.4
34			7.429	84.647	0.472	0.465	89.4			6.929	83.589	0.567	0.539	92.3
35			8.240	95.664	0.936	0.932	97.5			7.206	89.449	0.832	0.805	97.3

B.2 Model Assumption

Since the correlation between $\phi(x)$ and response variables y may be non-linear, we need to make further assumptions and approximation. Let each y be independently generated from a unknown distribution: $p(y|\Phi,f)$. Assume this distribution is unimodal and the mode is denoted by μ, we can take Taylor expansion of log likelihood at the mode

$$\log p\big(y|\phi(x),f\big) \approx \log p\big(\mu\big|\phi(x),f\big) - \frac{1}{2}(y-\mu)^\top \Lambda (y-\mu)$$

where $\Lambda = -\nabla_y \nabla_y \log p(y|\phi(x),f)\big|_{y=\mu}$. The above transformation is the Laplace approximation [51] and the quadratic term implies the rationality of the Gaussian approximation. Similar to You et al. [91], the top model over a learned feature extractor ϕ is approximated with a linear model:

$$y = \mathbf{w}^\top \phi(x) + \epsilon, \quad y \in \mathbb{R}, \mathbf{w} \in \mathbb{R}^d, \epsilon \in \mathbb{R},$$

where ϵ is a Gaussian noise with variance β^{-1}. We assume the prior distribution of the weights $\mathbf{w}$ is a zero-mean isotropic Gaussian distribution governed by a hyperparameter α:

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \alpha^{-1}\mathbb{I}_d) \quad \text{or} \quad p(\mathbf{w};\alpha) = \Big(\frac{\alpha}{2\pi}\Big)^{\frac{d}{2}} \exp\Big(-\frac{\alpha}{2}\mathbf{w}^\top \mathbf{w}\Big)$$

and the conditional distribution of the target variable y given $\phi(x)$ is a Gaussian distribution:

$$y\big|\phi(x),\mathbf{w} \sim \mathcal{N}(\mathbf{w}^\top \phi(x), \beta^{-1}) \quad \text{or} \quad p\big(y\big|\phi(x),\mathbf{w};\beta\big) = \Big(\frac{\beta}{2\pi}\Big)^{\frac{1}{2}} \exp\bigg(-\frac{\beta}{2}\big(y-\mathbf{w}^\top\phi(x)\big)^2\bigg).$$

Recall the notations $\mathbf{y}_i$, Φ_i, $\mathbf{y}_{-i}$ and Φ_{-i} in Appendix B.1. Then we have

$$\mathbf{y}_i|\Phi_i,\mathbf{w} \sim \mathcal{N}(\Phi_i\mathbf{w}, \beta^{-1}\mathbb{I}_{n_i}) \quad \text{and} \quad \mathbf{y}_{-i}|\Phi_{-i},\mathbf{w} \sim \mathcal{N}(\Phi_{-i}\mathbf{w}, \beta^{-1}\mathbb{I}_{n-n_i}).$$

In the next section, we present the details of estimating the two hyperparamters α and β. Appendix B.4 shows how to compute the conditional density $p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i})$ and $p(\Phi_i|\Phi_{-i})$ in the proposed metric (3).

Table 7: The ranking scores and fine-tuning accuracy for DomainNet dataset. The numbering in the first column corresponds to a pre-trained model from Table 3. The numbers in each subsequent column represent the scores assigned by a ranking metric to the PTMs. The last column displays the accuracy of each model after fine-tuning. Empty cells represent models for which ranking is not feasible.

Model	DomainNet						
Number	LEEP	NCE	H-Score	kNN	LogME	ZooD	Acc.
1	-4.083	-3.972	51.822	24.387	1.590	1.229	31.1
2	-3.946	-3.898	58.350	26.811	1.601	1.237	32.6
3	-4.033	-3.963	50.728	24.933	1.588	1.228	31.3
4	-3.984	-3.943	45.158	23.998	1.566	1.204	32.2
5	-3.989	-3.931	48.664	25.178	1.569	1.207	33.5
6	-4.646	-4.287	31.525	19.208	1.560	1.211	24.2
7	-3.999	-3.981	49.943	23.852	1.588	1.238	30.3
8	-4.172	-4.059	32.807	21.075	1.561	1.208	27.9
9	-4.177	-3.833	47.122	25.990	1.584	1.225	34.2
10	-3.768	-3.694	58.857	25.956	1.603	1.250	34.7
11	-4.063	-3.829	46.212	24.848	1.586	1.231	35.3
12	-3.914	-3.769	56.918	26.283	1.602	1.240	37.4
13	-4.127	-3.965	50.865	24.040	1.588	1.225	31.8
14	-4.252	-4.037	48.624	21.554	1.584	1.224	30.8
15			52.079	20.940	1.591	1.211	27.1
16			54.303	17.481	1.597	1.179	12.7
17			30.438	8.729	1.556	1.113	4.1
18			33.129	9.266	1.560	1.117	4.5
19			47.827	17.507	1.584	1.200	25.4
20			48.762	16.188	1.587	1.174	15.1
21			51.271	15.744	1.591	1.191	18.5
22			47.734	16.392	1.583	1.203	23.1
23	-4.078	-3.992	28.905	22.296	1.558	1.198	29.7
24	-3.787	-3.793	64.463	27.011	1.613	1.233	38.3
25	-3.788	-3.743	64.207	28.979	1.614	1.250	35.7
26	-3.661	-3.685	70.961	30.872	1.626	1.260	38.1
27	-3.841	-3.748	35.255	27.955	1.569	1.215	35.9
28	-3.426	-3.430	82.151	35.589	1.648	1.282	46.3
29	-3.413	-3.380	83.818	38.643	1.654	1.300	44.7
30	-3.229	-3.224	98.610	42.285	1.687	1.328	48.2
31	-3.646	-3.376	73.872	35.363	1.635	1.277	48.8
32	-5.639	-5.096	14.577	5.968	1.536	1.178	10.6
33	-4.226	-3.908	50.099	27.670	1.593	1.232	34.1
34			43.703	16.713	1.565	1.201	15.9
35			54.259	49.147	1.601	1.259	56.2

B.3 Parameter Estimation

If we introduce a uniform prior distribution over α and β, the posterior distribution for α and β is

$$p(\alpha,\beta|\mathbf{y}_{-i},\Phi_{-i}) = \frac{p(\alpha,\beta,\mathbf{y}_{-i},\Phi_{-i})}{p(\mathbf{y}_{-i},\Phi_{-i})} \propto p(\alpha,\beta,\mathbf{y}_{-i},\Phi_{-i}) = p(\mathbf{y}_{-i},\Phi_{-i}|\alpha,\beta)p(\alpha,\beta),$$

where the prior distribution $p(\alpha,\beta)$ is assumed to be a uniform distribution over α and β. Then the values of $\hat{\alpha}$ and $\hat{\beta}$ are obtained by maximizing the density function $p(\mathbf{y}_{-i},\Phi_{-i}|\alpha,\beta)$, which is also the model evidence over $\{\mathbf{y}_{-i},\Phi_{-i}\}$. The density function $p(\mathbf{y}_{-i},\Phi_{-i}|\alpha,\beta)$ is obtained by integrating over $\mathbf{w}$:

$$\begin{aligned}
p\big(\mathbf{y}_{-i},\Phi_{-i}\big|\alpha,\beta\big) &= \int_{\mathbf{w}} p\big(\mathbf{y}_{-i},\Phi_{-i}\big|\mathbf{w},\beta\big)p\big(\mathbf{w}\big|\alpha\big)\mathrm{d}\mathbf{w} \\
&= \int_{\mathbf{w}} p\big(\mathbf{y}_{-i}\big|\Phi_{-i},\mathbf{w},\beta\big)p\big(\Phi_{-i}\big|\mathbf{w},\beta\big)p\big(\mathbf{w}\big|\alpha\big)\mathrm{d}\mathbf{w} \\
&= \int_{\mathbf{w}} p\big(\mathbf{y}_{-i}\big|\Phi_{-i},\mathbf{w},\beta\big)p\big(\mathbf{w}\big|\alpha\big)\mathrm{d}\mathbf{w} \times p(\Phi_{-i}) \\
&\propto \int_{\mathbf{w}} p\big(\mathbf{y}_{-i}\big|\Phi_{-i},\mathbf{w},\beta\big)p\big(\mathbf{w}\big|\alpha\big)\mathrm{d}\mathbf{w}.
\end{aligned}$$

According to the model assumptions in Appendix B.2:

$$\mathbf{y}_{-i}|\Phi_{-i},\mathbf{w} \sim \mathcal{N}(\Phi_{-i}\mathbf{w},\beta^{-1}\mathbb{I}_{n-n_i}) \quad \text{and} \quad \mathbf{w} \sim \mathcal{N}(\mathbf{0},\alpha^{-1}\mathbb{I}_d),$$

then the likelihood function of α and β is

$$\begin{aligned} L(\alpha,\beta) &= \int_{\mathbf{w}} p(\mathbf{y}_{-i}|\Phi_{-i},\mathbf{w},\beta)p(\mathbf{w}|\alpha)\mathrm{d}\mathbf{w} \\ &= \left(\frac{\beta}{2\pi}\right)^{\frac{n-n_i}{2}}\left(\frac{\alpha}{2\pi}\right)^{\frac{d}{2}}\int_{\mathbf{w}}\exp\left(-\frac{\beta}{2}(\mathbf{y}_{-i}-\Phi_{-i}\mathbf{w})^\top(\mathbf{y}_{-i}-\Phi_{-i}\mathbf{w})-\frac{\alpha}{2}\mathbf{w}^\top\mathbf{w}\right)\mathrm{d}\mathbf{w} \\ &= \left(\frac{\beta}{2\pi}\right)^{\frac{n-n_i}{2}}\left(\frac{\alpha}{2\pi}\right)^{\frac{d}{2}}\int_{\mathbf{w}}\exp(-E(\mathbf{w}))\mathrm{d}\mathbf{w}, \end{aligned}$$

where $E(\mathbf{w})$ is the energy function of $\mathbf{w}$, i.e.

$$E(\mathbf{w})=\frac{\beta}{2}(\mathbf{y}_{-i}-\Phi_{-i}\mathbf{w})^\top(\mathbf{y}_{-i}-\Phi_{-i}\mathbf{w})+\frac{\alpha}{2}\mathbf{w}^\top\mathbf{w}.$$

Given $\mathbf{y}_{-i}$ and Φ_{-i}, then the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}|\mathbf{y}_{-i},\Phi_{-i},\alpha,\beta)\sim\mathcal{N}\left(\mathbf{w}|\mathbf{m}_{-i},\mathbf{A}_{-i}^{-1}\right),$$

where

$$\mathbf{m}_{-i}=\beta\mathbf{A}_{-i}^{-1}\Phi_{-i}^\top\mathbf{y}_{-i},\quad \mathbf{A}_{-i}=\alpha\mathbb{I}_d+\beta\Phi_{-i}^\top\Phi_{-i}.$$

Notice that

$$\begin{aligned} E(\mathbf{w}) &= \frac{\beta}{2}\mathbf{w}^\top\Phi_{-i}^\top\Phi_{-i}\mathbf{w}+\frac{\alpha}{2}\mathbf{w}^\top\mathbf{w}-\beta\mathbf{y}_{-i}^\top\Phi_{-i}\mathbf{w}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i} \\ &= \frac{1}{2}\mathbf{w}^\top(\beta\Phi_{-i}^\top\Phi_{-i}+\alpha\mathbb{I}_d)\mathbf{w}-\beta\mathbf{y}_{-i}^\top\Phi_{-i}\mathbf{w}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i} \\ &= \frac{1}{2}\mathbf{w}^\top\mathbf{A}_{-i}\mathbf{w}-\beta\mathbf{y}_{-i}^\top\Phi_{-i}\mathbf{A}_{-i}^{-1}\mathbf{A}_{-i}\mathbf{w}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i} \\ &= \frac{1}{2}\mathbf{w}^\top\mathbf{A}_{-i}\mathbf{w}-\mathbf{m}_{-i}^\top\mathbf{A}_{-i}\mathbf{w}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i}. \end{aligned}$$

Then we have $E(\mathbf{m}_{-i})=-\frac{1}{2}\mathbf{m}_{-i}^\top\mathbf{A}_{-i}\mathbf{m}_{-i}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i}$. We rewrite $\mathbf{w}=\mathbf{w}-\mathbf{m}_{-i}+\mathbf{m}_{-i}$ and obtain that

$$\frac{1}{2}\mathbf{w}^\top\mathbf{A}_{-i}\mathbf{w} = \frac{1}{2}(\mathbf{w}-\mathbf{m}_{-i})^\top\mathbf{A}_{-i}(\mathbf{w}-\mathbf{m}_{-i})-\frac{1}{2}\mathbf{m}_{-i}^\top\mathbf{A}_{-i}\mathbf{m}_{-i}+\mathbf{m}_{-i}^\top\mathbf{A}_{-i}\mathbf{w}.$$

Therefore,

$$\begin{aligned} E(\mathbf{w}) &= \frac{1}{2}(\mathbf{w}-\mathbf{m}_{-i})^\top\mathbf{A}_{-i}(\mathbf{w}-\mathbf{m}_{-i})-\frac{1}{2}\mathbf{m}_{-i}^\top\mathbf{A}_{-i}\mathbf{m}_{-i}+\frac{\beta}{2}\mathbf{y}_{-i}^\top\mathbf{y}_{-i} \\ &= E(\mathbf{m}_{-i})+\frac{1}{2}(\mathbf{w}-\mathbf{m}_{-i})^\top\mathbf{A}_{-i}(\mathbf{w}-\mathbf{m}_{-i}). \end{aligned}$$

Then we have

$$\begin{aligned} \log L(\alpha,\beta) &= \frac{n-n_i}{2}\log\beta+\frac{d}{2}\log\alpha-\frac{n-n_i}{2}\log(2\pi)-E(\mathbf{m}_{-i})-\frac{1}{2}\log|\mathbf{A}_{-i}| \qquad (4) \\ &= \frac{n-n_i}{2}\log\beta+\frac{d}{2}\log\alpha-\frac{n-n_i}{2}\log(2\pi)-\frac{\beta}{2}\left\|\mathbf{y}_{-i}-\Phi_{-i}\mathbf{m}_{-i}\right\|^2-\frac{\alpha}{2}\|\mathbf{m}_{-i}\|^2-\frac{1}{2}\log|\mathbf{A}_{-i}|. \end{aligned}$$

and obtain $\hat{\alpha}$ and $\hat{\beta}$ by maximizing $\log L(\alpha,\beta)$, i.e.,

$$\hat{\alpha},\hat{\beta}=\underset{\alpha,\beta}{\mathrm{argmax}}\log L(\alpha,\beta).$$

We can find that the objective function here is the same to Eq.(2) in You et al. [91]. Then we use the fix-point iteration algorithm [91, 90]. The detailed inference procedure are presented as follows.

Let λ_i and $\mathbf{v}_i$ be the i-th eigenvalue and eigenvector of the matrix $\beta\Phi_{-i}^\top\Phi_{-i}$. That is $(\beta\Phi_{-i}^\top\Phi_{-i})\mathbf{v}_i=\lambda_i\mathbf{v}_i$. Then we have

$$|\mathbf{A}_{-i}|=|\alpha\mathbb{I}_d+\beta\Phi_{-i}^\top\Phi_{-i}|=\prod_{i=1}^{d}(\alpha+\lambda_i).$$

The stationary points of $\log L(\alpha,\beta)$ with respect to α satisfy

$$\begin{aligned}
&\frac{d}{2\alpha}-\frac{1}{2}\|\mathbf{w}\|^2-\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}\alpha}\log\left(\prod_{i=1}^{d}(\alpha+\lambda_i)\right)=0\\
\Leftrightarrow\quad & d-\sum_{i=1}^{d}\frac{\alpha}{\alpha+\lambda_i}=\alpha\|\mathbf{w}\|^2\\
\Leftrightarrow\quad & \alpha=\frac{\gamma}{\|\mathbf{w}\|^2}\quad\text{with}\quad\gamma=\sum_{i=1}^{d}\frac{\lambda_i}{\alpha+\lambda_i}.
\end{aligned}$$

Notice that the eigenvalues λ_i are proportional to β. Hence $\mathrm{d}\lambda_i/\mathrm{d}\beta=\lambda_i/\beta$. Then the stationary points of $\log L(\alpha,\beta)$ with respect to β satisfy

$$\begin{aligned}
&\frac{n-n_i}{2\beta}-\frac{1}{2}\left\|\mathbf{y}_{-i}-\Phi_{-i}\mathbf{m}_{-i}\right\|^2-\frac{1}{2\beta}\sum_{i=1}^{d}\frac{\lambda_i}{\alpha+\lambda_i}=0\\
\Leftrightarrow\quad & \frac{1}{\beta}=\frac{1}{n-n_i-\gamma}\left\|\mathbf{y}_{-i}-\Phi_{-i}\mathbf{m}_{-i}\right\|^2.
\end{aligned}$$

B.4 Computing Metric

In this section, we present the details of computing the covariate shift $p(\Phi_i|\Phi_{-i})$ and the correlation shift $p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i})$. Then we can plug these two quantities into (3) to compute the proposed metric.

Covariate shift. Leaving the i-th domain out, we compute the density $p\big(\Phi_i\big|\Phi_{-i}\big)$ to check whether the learned feature $\phi(x)$ is stable such that the distribution shift between Φ_i and Φ_{-i} is not significant. We approximate the distribution of $\phi(x)$ with a Gaussian distribution $\mathcal{N}(\mu_\phi,\Sigma_\phi)$ and empirically estimate the parameters μ_ϕ and Σ_ϕ from the training inputs $\Phi_{-i}\in\mathbb{R}^{(n-n_i)\times d}$. That is,

$$\hat{\mu}_\phi=\frac{1}{n-n_i}\Phi_{-i}^\top\mathbb{1}_{n-n_i}\quad\hat{\Sigma}_\phi=\frac{1}{n-n_i}(\Phi_{-i}-\mathbb{1}_{n-n_i}\hat{\mu}_\phi^\top)^\top(\Phi_{-i}-\mathbb{1}_{n-n_i}\hat{\mu}_\phi^\top),$$

where $\mathbb{1}_{n-n_i}$ is a $(n-n_i)$-length one vector. Then we compute the density of Φ_i according to $\mathcal{N}(\hat{\mu}_\phi,\hat{\Sigma}_\phi)$:

$$\begin{aligned}
p(\Phi_i|\Phi_{-i}) &= p(\Phi_i|\hat{\mu}_\phi,\hat{\Sigma}_\phi)=\prod_{j=1}^{n_i}\sqrt{\frac{1}{(2\pi)^d|\hat{\Sigma}_\phi|}}\exp\left(-\frac{1}{2}(\phi(x_{ij})-\hat{\mu}_\phi)^\top\hat{\Sigma}_\phi^{-1}(\phi(x_{ij})-\hat{\mu}_\phi)\right).\\
&= (2\pi)^{-\frac{(n-n_i)d}{2}}|\hat{\Sigma}_\phi|^{-\frac{n-n_i}{2}}\exp\left(-\frac{1}{2}\mathrm{trace}\big\{(\Phi_{-i}-\mathbb{1}_{n-n_i}\hat{\mu}_\phi^\top)\hat{\Sigma}_\phi^{-1}(\Phi_{-i}-\mathbb{1}_{n-n_i}\hat{\mu}_\phi^\top)^\top\big\}\right).
\end{aligned}$$

Correlation shift. Given $\hat{\alpha}$ and $\hat{\beta}$, we have

$$p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i};\hat{\alpha},\hat{\beta})=\frac{p(\mathbf{y}_i,\mathbf{y}_{-i}|\Phi_i,\Phi_{-i};\hat{\alpha},\hat{\beta})}{p(\mathbf{y}_{-i}|\Phi_i,\Phi_{-i};\hat{\alpha},\hat{\beta})}=\frac{p(\mathbf{y}_i,\mathbf{y}_{-i}|\Phi_i,\Phi_{-i};\hat{\alpha},\hat{\beta})}{p(\mathbf{y}_{-i}|\Phi_{-i};\hat{\alpha},\hat{\beta})}.\tag{5}$$

We write $\hat{\mathbf{m}}_{-i}=\hat{\beta}\hat{\mathbf{A}}_{-i}^{-1}\Phi_{-i}^\top\mathbf{y}_{-i}$ and $\hat{\mathbf{A}}_{-i}=\hat{\alpha}\mathbb{I}_d+\hat{\beta}\Phi_{-i}^\top\Phi_{-i}$. According to (4),

$$\begin{aligned}
\log p(\mathbf{y}_{-i}|\Phi_{-i};\hat{\alpha},\hat{\beta}) &= \frac{n-n_i}{2}\log\hat{\beta}+\frac{d}{2}\log\hat{\alpha}-\frac{n-n_i}{2}\log(2\pi)\\
&\quad-\frac{\hat{\beta}}{2}\left\|\mathbf{y}_{-i}-\Phi_{-i}\hat{\mathbf{m}}_{-i}\right\|^2-\frac{\hat{\alpha}}{2}\|\hat{\mathbf{m}}_{-i}\|^2-\frac{1}{2}\log|\hat{\mathbf{A}}_{-i}|.
\end{aligned}\tag{6}$$

To proceed further, we denote

$$\mathbf{y}=(\mathbf{y}_i^\top,\mathbf{y}_{-i}^\top)^\top\in\mathbb{R}^n,\quad\Phi=(\Phi_i^\top,\Phi_{-i}^\top)^\top\in\mathbb{R}^{n\times d},\quad\hat{\mathbf{m}}=\hat{\beta}\hat{\mathbf{A}}^{-1}\Phi^\top\mathbf{y},\quad\hat{\mathbf{A}}=\hat{\alpha}\mathbb{I}_d+\hat{\beta}\Phi^\top\Phi.$$

Similar to (4), we have

$$\begin{aligned}
\log p(\mathbf{y}|\Phi;\hat{\alpha},\hat{\beta}) &= \log p(\mathbf{y}_i,\mathbf{y}_{-i}|\Phi_i,\Phi_{-i};\hat{\alpha},\hat{\beta})\\
&= \frac{n}{2}\log\hat{\beta}+\frac{d}{2}\log\hat{\alpha}-\frac{n}{2}\log(2\pi)-\frac{\hat{\beta}}{2}\left\|\mathbf{y}-\Phi\hat{\mathbf{m}}\right\|^2-\frac{\hat{\alpha}}{2}\|\hat{\mathbf{m}}\|^2-\frac{1}{2}\log|\hat{\mathbf{A}}|.
\end{aligned}\tag{7}$$

Plugging (6) and (7) into (3), we obtain the value of the proposed metric.

Remark. Given $\mathbf{y}_{-i}$, Φ_{-i}, $\hat{\alpha}$ and $\hat{\beta}$, the posterior distribution of $\mathbf{w}$ is

$$p(\mathbf{w}|\mathbf{y}_{-i},\Phi_{-i},\hat{\alpha},\hat{\beta}) \sim \mathcal{N}\left(\mathbf{w}|\hat{\mathbf{m}}_{-i},\hat{\mathbf{A}}_{-i}^{-1}\right).$$

Further,

$$p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i};\hat{\alpha},\hat{\beta}) = \int_{\mathbf{w}} p(\mathbf{y}_i|\Phi_i,\mathbf{w};\hat{\beta}) p(\mathbf{w}|\mathbf{y}_{-i},\Phi_{-i};\hat{\alpha},\hat{\beta}) \mathrm{d}\mathbf{w}.$$

By calculating the integral, we can deduce

$$\mathbf{y}_i \big| \Phi_i,\mathbf{y}_{-i},\Phi_{-i} \sim \mathcal{N}\big(\Phi_i \hat{\mathbf{m}}_{-i}, \hat{\beta}^{-1}\mathbb{I}_{n_i} + \Phi_i \hat{\mathbf{A}}_{-i}^{-1} \Phi_i^\top\big).$$

Therefore we can also use this distribution to calculate $p(\mathbf{y}_i|\Phi_i,\mathbf{y}_{-i},\Phi_{-i})$ directly. Throughout this paper, we use the formula (5) to calculate the correlation shift.

B.5 Cross-Domain Validation Selects Invariant Features

To justify our proposed selection method, and provide more intuition, we conduct explicit analysis in a linear regression setting. Despite the over-simplification, it does reflect the essence of our approach. From this base case, adaptions to more complicated and realistic assumptions can be made.

Data Assumption Suppose we have data in different domains with domain invariant and domain-specific features, with respect to the response variable y. Denote the set of invariant features to be iv, which are assumed to be unit-norm and orthogonal to each other. Without loss of generality, let data in domain $\mathcal{D}$ be $\boldsymbol{x} = (\boldsymbol{x}_{iv}, \boldsymbol{x}_{\mathcal{D}})$ where $\boldsymbol{x}_{iv} \in \mathbb{R}^{d^*}$ denotes the domain invariant features and $\boldsymbol{x}_{\mathcal{D}} \in \mathbb{R}^{d-d^*}$ denotes domain specific ones. Let $\boldsymbol{x}_{iv}$ be fixed. The domain specific features can have non-zero correlation with $\boldsymbol{x}_{iv}$ such that

$$\boldsymbol{x}_{\mathcal{D}} = \boldsymbol{x}_{iv} \cdot \boldsymbol{A}_{\mathcal{D}} + \boldsymbol{e}_{\mathcal{D}},$$

where $\boldsymbol{A}_{\mathcal{D}} \in \mathbb{R}^{d^* \times (d-d^*)}$, and $\boldsymbol{e}_{\mathcal{D}} \sim N(0, s^2 \boldsymbol{I}_{d-d^*})$. For different domains, assume the correlation to be independently random, i.e., $\boldsymbol{A}_{\mathcal{D}}$'s are i.i.d. matrices with independent entries with mean 0 and variance 1. Given the features $\boldsymbol{x}$, assume the response y only depends on $\boldsymbol{x}_{iv}$ such that

$$y = \boldsymbol{x}_{iv} \cdot \beta_{iv} + \epsilon = \boldsymbol{x} \cdot \beta + \epsilon,$$

where $\beta = (\beta_{iv}, \beta^{\mathcal{D}})$ with $\beta^{\mathcal{D}} = \mathbf{0}$ and ϵ follows $N(0, \sigma^2)$.

Model Assumption Let the model candidates be linear models fitted to different subsets of the features and there are in total 2^d different combinations. Denote the fitted parameters to be $\hat{\beta} \in \mathbb{R}^d$ with only the selected dimensions being non-zero. Let the selection be ϕ, which is a subset of $\{1,...,d\}$. We want to show that our proposed statistics, in the cross-validated fashion, will prefer the optimal one with $\phi = iv$. The optimality is in the sense that it achieves the best goodness-of-fit, measured by the square loss.

More Notations Let $(\boldsymbol{X}, \boldsymbol{y}), (\tilde{\boldsymbol{X}}, \tilde{\boldsymbol{y}})$ be independent datasets in two domains to be cross validated. For any vector (matrix), we use subscript to denote part of it with selected rows (columns). For instance, a model candidates with feature dimensions ϕ will only fit $\boldsymbol{y} \sim \boldsymbol{X}_\phi$ and the resulting $\hat{\beta}$ will only be nonzero on $\hat{\beta}_\phi$. For a set ϕ, denote $|\phi|$ be to its cardinality and $\bar{\phi}$ to be its complement.

In our proposed test statistics, there are two terms to be assessed. The first term is essentially the goodness-of-fit of $\tilde{\boldsymbol{y}}$ and $\tilde{\boldsymbol{X}}_\phi \cdot \hat{\beta}_\phi$, which is of critical importance for selecting the invariance and consistent features cross different domains. The second term can be seen as some regularization. In this section, we will focus on the first term and to make things really simple, we consider expected l_2 loss as the measure for goodness-of-fit.

The estimated $\hat{\beta}$ can be explicitly written as

$$\hat{\beta}_\phi = (\boldsymbol{X}_\phi^\top \boldsymbol{X}_\phi)^{-1} \boldsymbol{X}_\phi^\top \boldsymbol{y} \in \mathbb{R}^{|\phi|}.$$

Given $\boldsymbol{y}=\boldsymbol{X}\beta+\boldsymbol{\epsilon}$, we can write

$$\boldsymbol{X}\beta=\boldsymbol{X}_{\phi}\beta_{\phi}+\boldsymbol{X}_{\bar{\phi}}\beta_{\bar{\phi}}.$$

Thus,

$$\begin{aligned}\hat{\beta}_{\phi}&=\beta_{\phi}+(\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\bar{\phi}}\beta_{\bar{\phi}}+(\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}\boldsymbol{X}_{\phi}^{\top}\boldsymbol{\epsilon}\\&=\beta_{\phi}+(\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}\boldsymbol{X}_{\phi}^{\top}\boldsymbol{\epsilon}.\end{aligned}\tag{8}$$

The expected l_2 loss can be expressed as

$$\begin{aligned}&\mathbb{E}_{\epsilon,\tilde{\epsilon},e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{y}}-\tilde{\boldsymbol{X}}_{\phi}\cdot\hat{\beta}_{\phi}\|^2\right)\\=&\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{X}}\cdot\beta-\tilde{\boldsymbol{X}}_{\phi}\cdot\hat{\beta}_{\phi}\|^2\right)+n\sigma^2\\=&\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{X}}_{iv\cap\phi}\beta_{iv\cap\phi}+\tilde{\boldsymbol{X}}_{iv\setminus\phi}\beta_{iv\setminus\phi}-\tilde{\boldsymbol{X}}_{\phi\cap iv}\cdot\hat{\beta}_{\phi\cap iv}-\tilde{\boldsymbol{X}}_{\phi\setminus iv}\cdot\hat{\beta}_{\phi\setminus iv}\|^2\right)+n\sigma^2\\=&\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{X}}_{iv\cap\phi}(\beta_{iv\cap\phi}-\hat{\beta}_{iv\cap\phi})+\tilde{\boldsymbol{X}}_{iv\setminus\phi}\beta_{iv\setminus\phi}-\tilde{\boldsymbol{X}}_{\phi\setminus iv}\cdot\hat{\beta}_{\phi\setminus iv}\|^2\right)+n\sigma^2\\=&\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{X}}_{iv\cap\phi}\left((\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}\boldsymbol{X}_{\phi}^{\top}\boldsymbol{\epsilon}\right)_{iv\cap\phi}+\tilde{\boldsymbol{X}}_{iv\setminus\phi}\beta_{iv\setminus\phi}-\tilde{\boldsymbol{X}}_{\phi\setminus iv}\cdot\hat{\beta}_{\phi\setminus iv}\|^2\right)+n\sigma^2\\:=&\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\|I_1+I_2+I_3\|^2\right)+n\sigma^2.\end{aligned}$$

I_1 accounts for the variance in estimating the selected invariance features. I_2 is non-random and accounts for the error from unselected invariance features. I_3 accounts the error from wrongly selected features. Easy to verify that $\mathbb{E}(I_1)=\mathbb{E}(I_3)=0$ and $\mathbb{E}(I_1I_3)=0$, since $\hat{\beta}$ is independent with $\tilde{A},\tilde{e}$, which are both mean zero.

$$\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}(\|I_1\|^2)=\sigma^2\mathbb{E}_{e,A}\mathrm{tr}\left((\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}_{iv\cap\phi}\right)$$

For I_3, we can further write

$$\begin{aligned}\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}(\|I_3\|^2)&=\mathbb{E}_{\epsilon,e,A,\tilde{e},\tilde{A}}\left(\hat{\beta}_{\phi\setminus iv}^{\top}\tilde{\boldsymbol{X}}_{\phi\setminus iv}^{\top}\tilde{\boldsymbol{X}}_{\phi\setminus iv}\cdot\hat{\beta}_{\phi\setminus iv}\right)\\&=\mathbb{E}_{\epsilon,e,A}\left(\|\hat{\beta}_{\phi\setminus iv}\|^2\right)\mathbb{E}_{\tilde{e},\tilde{A}}\mathrm{tr}\left(\tilde{\boldsymbol{X}}_{\phi\setminus iv}^{\top}\tilde{\boldsymbol{X}}_{\phi\setminus iv}\right)\\&=\mathbb{E}_{\epsilon,e,A}\left(\|\hat{\beta}_{\phi\setminus iv}\|^2\right)\left(\mathbb{E}_{\tilde{A}}\mathrm{tr}\left(\tilde{\boldsymbol{A}}_{\phi\setminus iv}^{\top}\tilde{\boldsymbol{A}}_{\phi\setminus iv}\right)+n|\phi\setminus iv|s^2\right)\\&=n(1+s^2)|\phi\setminus iv|\cdot\mathbb{E}_{\epsilon,e,A}\left(\|\hat{\beta}_{\phi\setminus iv}\|^2\right)\end{aligned}$$

Therefore,

$$\begin{aligned}&\mathbb{E}_{\epsilon,\tilde{\epsilon},e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{y}}-\tilde{\boldsymbol{X}}_{\phi}\cdot\hat{\beta}\|^2\right)\\=&\sigma^2\mathbb{E}_{e,A}\mathrm{tr}\left((\boldsymbol{X}_{\phi}^{\top}\boldsymbol{X}_{\phi})^{-1}_{iv\cap\phi}\right)+\|\beta_{iv\setminus\phi}\|^2+n(1+s^2)|\phi\setminus iv|\cdot\mathbb{E}_{\epsilon,e,A}\left(\|\hat{\beta}_{\phi\setminus iv}\|^2\right)+n\sigma^2.\end{aligned}$$

If $\phi=iv$, the above quantity is minimized with $\mathbb{E}_{\epsilon,\tilde{\epsilon},e,A,\tilde{e},\tilde{A}}\left(\|\tilde{\boldsymbol{y}}-\tilde{\boldsymbol{X}}_{\phi}\cdot\hat{\beta}\|^2\right)=(n+d^*)\sigma^2$.

C Feature Selection in ZooD

In this section, we present more details about the PTMs ensemble and feature selection in Section 3.2. The top-ranked PTMs in Section 3.1 are preferred for solving the OoD generalization task. To further aggregate different PTMs, we consider assembling the features by using PTMs as feature extractors

$$\Phi = \left[\Phi^{(1)},...,\Phi^{(k)}\right],$$

where $\Phi^{(i)}$ is the i-th ranked feature extractor and $[\cdot]$ denotes the row concatenation operation. As we show in experiments, in most cases, using aggregated models can significantly outperform any single model. However, the rough ensemble will inevitably introduce more noise. According to the definition of OoD learnability proposed by Ye et al. [88], non-informative but invariant features from training domains may only bring some noise, and the accumulation of noise hurts learnability of the OoD generalization task. Therefore, we propose a Bayesian feature selection method based on the Gaussian linear framework in Section 3.1.

C.1 Bayesian Variable Selection

In the Bayesian literature, the variable selection problem can be efficiently solved by introducing, for each variable w_i, a binary mask $\mathrm{z}_i \in \{0,1\}$ [48, 17, 84, 87], which are given by Bernoulli distributions governed by probability coefficient $\boldsymbol{\pi}$. Let $\mathrm{z} = \{\mathrm{z}_i\}_{i=1}^d$ and

$$p(\mathrm{z};\boldsymbol{\pi}) = \prod_{i=1}^{d} p(\mathrm{z}_i) = \prod_{i=1}^{d} \pi_i^{\mathrm{z}_i}(1-\pi_i)^{1-\mathrm{z}_i}.$$

From a generative perspective, these masks determine whether the weight w_i is generated from a slab or a spike prior [37]. If $\mathrm{z}_i = 1$, then w_i will follow a slab prior with diffusing probability density; if $\mathrm{z}_i = 0$, w_i will have a spike prior with probability mass concentrated around 0, and thus should be discarded. Specifically, we assume

$$p(\mathrm{w}_i|\mathrm{z}_i,\alpha_{i,1},\alpha_{i,2}) = \begin{cases} \mathcal{N}(0,\alpha_{i,1}^{-1}) & \text{if } \mathrm{z}_i = 1; \\ \mathcal{N}(0,\alpha_{i,2}^{-1}) & \text{if } \mathrm{z}_i = 0. \end{cases}$$

Denote $\mathbf{w} = (\mathrm{w}_1,...,\mathrm{w}_d)^\top$ and $\alpha_{i,1}$ and $\alpha_{i,2}$ control the shape of the w_i distribution and should be reasonably large for $\alpha_{i,2}$. Conditioned on w_i, each data point y_n is assumed to be independently drawn from a linear model with mean $\mathbf{w}^\top \phi(x)$ and additional Gaussian noise with inverse variance β:

$$p(y_n \big| \phi(x_n),\mathbf{w};\beta) = \left(\frac{\beta}{2\pi}\right)^{\frac{1}{2}} \exp\left(-\frac{\beta}{2}\left(y_n - \mathbf{w}^\top \phi(x_n)\right)^2\right).$$

The model specification is completed by introducing conjugate Gamma priors over the inverse variance β and $\{\alpha_{i,1},\alpha_{i,2}\}_{i=1}^d$:

$$\alpha_{i,1} \sim \text{Gamma}(\nu_{i,1},\nu_{i,2}), \quad \alpha_{i,2} \sim \text{Gamma}(\nu_{i,3},\nu_{i,4}), \quad \beta \sim \text{Gamma}(\nu_{0,1},\nu_{0,2}).$$

Denote the set of Gamma prior parameters as $\boldsymbol{\nu} = \{\nu_{i,j}\}$ and all latent variables as

$$\boldsymbol{\xi} = \left\{\beta, \{\mathrm{w}_i,\mathrm{z}_i,\alpha_{i,1},\alpha_{i,2}\}_{i=1}^d\right\}.$$

Then the variable selection problem can be solved by estimating $\boldsymbol{\pi} = \{\pi_1,\pi_2,...,\pi_d\}$ with $\pi_i = p(\mathrm{z}_i = 1)$. We can find the maximum likelihood estimator of the probability coefficient $\boldsymbol{\pi}$ of Bernoulli masks and then screen the variables if π_i is smaller than the pre-defined threshold τ.

C.2 Variational EM Algorithm

Given the dataset $\{\mathbf{y},\Phi\}$, the maximum marginal likelihood estimator of $(\boldsymbol{\pi},\boldsymbol{\nu})$ is given by

$$\begin{aligned} \hat{\boldsymbol{\pi}},\hat{\boldsymbol{\nu}} &= \underset{\boldsymbol{\pi},\boldsymbol{\nu}}{\text{argmax}} \log p(\mathbf{y}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) \\ &= \underset{\boldsymbol{\pi},\boldsymbol{\nu}}{\text{argmax}} \log \int_{\boldsymbol{\xi}} p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) \mathrm{d}\boldsymbol{\xi}. \end{aligned} \tag{9}$$

However, direct maximization of (9) is intractable due to the integration over $\boldsymbol{\xi}$. EM algorithm [67] might be a solution here. In the E-step, we compute the conditional expectation

$$\begin{aligned}
& \mathcal{L}(\boldsymbol{\pi},\boldsymbol{\nu};\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old}) \\
= \; & \mathbb{E}_{\boldsymbol{\xi}}\left[\log p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu})\middle|\mathbf{y},\Phi;\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old}\right] \\
= \; & \int \log p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) p(\boldsymbol{\xi}|\mathbf{y},\Phi;\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old}) \mathrm{d}\boldsymbol{\xi},
\end{aligned}$$

which involves inferring posterior $p(\boldsymbol{\xi}|\mathbf{y},\Phi;\boldsymbol{\pi},\boldsymbol{\nu})$. However this is not straightforward to obtain due to the complexity of our model setup. MCMC [58] is a common tool for this problem, but suffers from intensive computation, thus hard to extend to large scale data. We instead use approximate Bayesian inference in Section C.3.

In the M-step, we update $\boldsymbol{\pi}$ and $\boldsymbol{\nu}$ by maximizing the expectation

$$\boldsymbol{\pi}^{new},\boldsymbol{\nu}^{new} = \underset{\boldsymbol{\pi},\boldsymbol{\nu}}{\operatorname{argmax}} \mathcal{L}(\boldsymbol{\pi},\boldsymbol{\nu};\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old}).$$

By repeating the E and M steps, the estimator $(\boldsymbol{\pi}^{new},\boldsymbol{\nu}^{new})$ converges to an optimal solution. We show this method has satisfying performance for the underlying variable selection problems in synthetic data and the prevailing OoD dataset.

C.3 Variational Inference

In the E-Step, computation of $\mathbb{E}_{\boldsymbol{\xi}}\left[\log p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu})\middle|\mathbf{y},\Phi;\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old}\right]$ involves inferring posterior $p(\boldsymbol{\xi}|\mathbf{y},\Phi;\boldsymbol{\pi},\boldsymbol{\nu})$. However, due to the complexity of our model setup, no analytical form of the posterior distribution can be found. We instead approximate true posterior distribution by variational inference [13]. The main idea involves the introduction of a set of distributions Q, which should ideally be easy to compute and provide a good approximation to the true posterior distribution. We consider the following transformation of the marginal likelihood

$$\begin{aligned}
\ln p(\mathbf{y}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) &= \ln \int p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) d\boldsymbol{\xi} \\
&= \ln \int Q(\boldsymbol{\xi}) \frac{p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu})}{Q(\boldsymbol{\xi})} d\boldsymbol{\xi} \\
&\geq \int Q(\boldsymbol{\xi}) \ln \frac{p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu})}{Q(\boldsymbol{\xi})} d\boldsymbol{\theta} \\
&= \mathcal{L}(Q),
\end{aligned}$$

where $\mathcal{L}(Q)$ denotes the variational lower bound. The key point is that, through proper choice of Q distribution, $\mathcal{L}(Q)$ can be readily evaluated, and thus by maximizing the lower bound, we generally find the Q distribution, which is the best approximation within the considered family. Here we factorize Q over each latent variable, such that

$$Q(\boldsymbol{\xi};\boldsymbol{\pi},\boldsymbol{\nu}) = Q(\beta;\tilde{\nu}_{0,1},\tilde{\nu}_{0,2}) \prod_{i=1}^{d} \Big[Q(\mathrm{z}_i;\tilde{\pi}_i) Q(\mathrm{w}_i;m_i,\lambda_i^{-1}) Q(\alpha_{i,1};\tilde{\nu}_{i,1},\tilde{\nu}_{i,2}) Q(\alpha_{i,2};\tilde{\nu}_{i,3},\tilde{\nu}_{i,4}) \Big],$$

which holds for classic mean-field family [12]. By denoting $\{\boldsymbol{m},\boldsymbol{\lambda},\tilde{\boldsymbol{\pi}}\} = \{m_i,\lambda_i,\pi_i\}_{i=1}^d$ and $\tilde{\boldsymbol{\nu}} = \{\tilde{\nu}_{i,j}\}$, an optimization-free form over all possible Q has been established, which can lead to minimization of KL divergence between variational distribution $Q(\boldsymbol{\xi})$ and true posterior $p(\boldsymbol{\xi}|\mathbf{y},\Phi;\boldsymbol{\pi},\boldsymbol{\nu})$

$$Q_i^*(\boldsymbol{\xi}_i) = \frac{\exp \mathbb{E}_{\boldsymbol{\xi}_{k\neq i} \sim Q^*(\boldsymbol{\xi}_{k\neq i})} \ln p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu})}{\int \exp \mathbb{E}_{\boldsymbol{\xi}_{k\neq i} \sim Q^*(\boldsymbol{\xi}_{k\neq i})} \ln p(\mathbf{y},\boldsymbol{\xi}|\Phi;\boldsymbol{\pi},\boldsymbol{\nu}) d\boldsymbol{\xi}_i}.$$

For models in conjugate families, the optimal $Q^*(\boldsymbol{\xi}_i)$ has the same form as its prior distribution. We then establish the optimization step for arbitrary variational parameters set $\{\boldsymbol{m},\boldsymbol{\lambda},\tilde{\boldsymbol{\nu}},\tilde{\boldsymbol{\pi}}\}$ to approach the true posterior:

$$m_i = f_m(\tilde{\pi}_i,\boldsymbol{m},\tilde{\boldsymbol{\nu}}) = \left(\sum_{n=1}^{N} x_{n,i}^2 \mathbb{E}[\beta] + \tilde{\pi}_i \mathbb{E}[\alpha_{i,1}] + (1-\tilde{\pi}_i) \mathbb{E}[\alpha_{i,2}] \right)^{-1} \cdot \left[\mathbb{E}[\beta] \cdot \sum_{n=1}^{N} x_{n,i} \left(\sum_{j\neq i}^{d-1} m_j \cdot x_{n,j} - y_n \right) \right],$$

$$\tilde{\pi}_i = f_{\pi_i}(\boldsymbol{m},\boldsymbol{\lambda},\tilde{\boldsymbol{\nu}}) = \frac{\exp\left\{\mathbb{E}\ln|\alpha_{i,1}| - \frac{1}{2}\mathbf{Tr}\left(\mathbb{E}[\alpha_{i,1}]\cdot[\mathbb{E}[\mathrm{w}_i^2]]\right) + \ln\pi_i\right\}}{\exp\left\{\mathbb{E}\ln|\alpha_{i,1}| + \mathbb{E}\ln|\alpha_{i,2}| - \frac{1}{2}\mathbf{Tr}[(\mathbb{E}[\alpha_{i,1}] + E[\alpha_{i,2}])\cdot[\mathbb{E}[\mathrm{w}_i^2]]] + \ln\pi_i + \ln(1-\pi_i)\right\}},$$

$$(\tilde{\nu}_{0,2})^{-1} = f_{\nu_{0,2}}(\boldsymbol{m},\boldsymbol{\lambda}) = \sum_{n=1}^{N} y_n^2 - 2\sum_{n=1}^{N}\left(\sum_{i=1}^{d} m_i\cdot x_{n,i}\right)\cdot y_n + \sum_{n=1}^{N}\sum_{i,j}^{d^2} x_{n,i}\cdot x_{nj}\left([\mathbb{E}[\mathrm{w}_i^2]\right) + \nu_{0,2}^{-1},$$

$$(\tilde{\nu}_{i,2})^{-1} = f_{\nu_{i,2}}(\boldsymbol{m},\boldsymbol{\lambda},\tilde{\boldsymbol{\pi}}) = \left(\mathbb{E}[\mathrm{w}_i^2]^{-1}\right)\cdot\tilde{\pi}_i + \nu_{i,2}^{-1}, \quad (\tilde{\nu}_{i,4})^{-1} = f_{\nu_{i,4}}(\boldsymbol{m},\boldsymbol{\lambda},\tilde{\boldsymbol{\pi}}) = \left(\mathbb{E}[\mathrm{w}_i^2]^{-1}\right)\cdot(1-\tilde{\pi}_i) + \nu_{i,4}^{-1},$$

$$\lambda_i = f_{\lambda}(\tilde{\boldsymbol{\nu}}) = \sum_{n=1}^{N} x_{n,i}^2\mathbb{E}[\beta] + \tilde{\pi}_i\mathbb{E}[\alpha_{i,1}] + (1-\tilde{\pi}_i)\mathbb{E}[\alpha_{i,2}],$$

$$\tilde{\nu}_{0,1} = f_{\nu_{0,1}}(n) = \nu_{0,1} + n, \quad \tilde{\nu}_{i,1} = f_{\nu_{i,1}}(\tilde{\boldsymbol{\pi}}) = \nu_{i,1} + \tilde{\pi}_i, \quad \tilde{\nu}_{i,3} = f_{\nu_{i,3}}(\tilde{\boldsymbol{\pi}}) = \nu_{i,3} + 1 - \tilde{\pi}_i,$$

where the variational expectations are given by

$$\mathbb{E}[\mathrm{w}_i^2] = m_i^2 + \lambda_i^{-1}, \quad \mathbb{E}[\beta] = \tilde{\nu}_{0,1}\cdot\tilde{\nu}_{0,2}, \quad \mathbb{E}[\alpha_{i,1}] = \tilde{\nu}_{i,1}\cdot\tilde{\nu}_{i,2}, \quad \mathbb{E}[\alpha_{i,2}] = \tilde{\nu}_{i,3}\cdot\tilde{\nu}_{i,4}, \tag{10}$$

$$\mathbb{E}\ln|\alpha_{i,1}| = \psi\left(\frac{\nu^{i,1}}{2}\right) + \ln 2 + \ln\left|\nu^{i,2}\right|, \quad \mathbb{E}\ln|\alpha_{i,2}| = \psi\left(\frac{\nu^{i,3}}{2}\right) + \ln 2 + \ln\left|\nu^{i,4}\right|. \tag{11}$$

Since the optimization steps for each variational parameter are mutually dependent, we can use coordinate gradient descent [13] starting by current $Q(\boldsymbol{\xi})^{t-1}$ from the last iteration. After one-step optimization, variational parameters of $Q(\boldsymbol{\xi})^t$ are used in computation of $\mathbb{E}_{\boldsymbol{\xi}\sim Q(\boldsymbol{\xi};\boldsymbol{\pi}^{old},\boldsymbol{\nu}^{old})^t}\left[\log p(\mathbf{y}|\Phi,\boldsymbol{\xi};\boldsymbol{\pi},\boldsymbol{\nu})\right]$, thus finishing E-step. During this procedure, the lower bound $\mathcal{L}(Q)$ will continuously increase until reaching its maximum value. Therefore, the value of $\mathcal{L}(Q)$ can be used as a useful indicator for convergence of algorithm [20].

C.4 Algorithm Details

The proposed model contains a set of prior hyper-parameters $\boldsymbol{\pi}$,$\boldsymbol{\nu}$, which is exactly what we want to estimate for feature screening. In Bayesian literature, hyper-parameter selection can be automated from data through a procedure named "ARD" [52]. The original "ARD" procedure proposes a selection based on the value of model evidence. However, in many cases including ours, this evidence is intractable. Fortunately, it's also feasible to use variational lower bound $\mathcal{L}(Q)$ as a substitute. Learning prior hyper-parameters $\boldsymbol{\pi}$,$\boldsymbol{\nu}$ leads to minimization of KL divergence. This can be rationalized by decomposition of $\mathcal{L}(Q)$:

$$\mathcal{L}(Q) = \mathbb{E}_{\boldsymbol{\xi}\sim Q(\boldsymbol{\xi})}\left[\log p(\mathbf{y}|\Phi,\boldsymbol{\xi};\boldsymbol{\pi},\boldsymbol{\nu})\right] = \mathbb{E}_{\boldsymbol{\xi}\sim Q(\boldsymbol{\xi})}\left[\log p(\mathbf{y}|\boldsymbol{\xi},\Phi)\right] - \mathrm{KL}(Q(\boldsymbol{\xi})||p(\boldsymbol{\xi};\boldsymbol{\pi},\boldsymbol{\nu})).$$

Thus by setting derivatives of each hyper-parameters with respect to $\mathcal{L}(Q)$ to 0, it's easy to see $\mathcal{L}(Q)$ is maximized when all hyper-parameters are set to posterior parameters:

$$\boldsymbol{\pi}^{new} = \tilde{\boldsymbol{\pi}}, \quad \boldsymbol{\nu}^{new} = \tilde{\boldsymbol{\nu}}.$$

However the proposed algorithm still suffers from heavy computational cost: Each iteration costs $\mathcal{O}(nd^2)$. Thus to relieve computation burden and memory usage, we leverage our method with stochastic approximation leading to the EM algorithm with stochastic variational inference [35]. In each iteration, we sample a random subset of entire data with size n^s. Fitting our algorithm over this subset for current iteration, we obtain a local optimal estimator denoted by $Q^s(\boldsymbol{\xi})$. In M-step these intermediate variational distributions by factorizing $Q^s(\boldsymbol{\xi})$ will be used to learn hyper-parameters $\boldsymbol{\pi}$ and $\boldsymbol{\nu}$ and simultaneously as the starting point for subsequent estimator in the next iteration. In the end, we successfully reduce the computation cost to $\mathcal{O}(n^s d^2)$ with $n^s \ll n$, while maintaining the guarantee of convergence to the global optimum [66]. In our experiments, we collect variational probabilities of $\{\tilde{\pi}_i\}_{i=1}^d$ from the last three runs and early-stop the algorithm if its difference with current probability is smaller than the pre-defined threshold ϵ or reaches the maximum iteration times. Variational EM algorithm for Bayesian feature selection is summarized in Algorithm 1. Note that we initialize $\boldsymbol{m}$ by linear regression and the initialization of $\tilde{\boldsymbol{\nu}}$ is set to $\boldsymbol{\nu}$.

In our experiments, we often deal with the multivariate case. If the underlying task involves multivariate regression or classification, i.e., $\boldsymbol{Y}\in\mathbb{R}^{n\times K}$, we can run the proposed EM algorithm on each dimension and take the union of all selected features. Therefore, our feature selection procedure can be used in almost all prevailing models and tasks.

Algorithm 1 Variational EM Algorithm for Bayesian Feature Selection

Input: The observed data $\boldsymbol{Y} \in \mathbb{R}^n, \boldsymbol{X} \in \mathbb{R}^{n \times d}$; Prior parameters $\boldsymbol{\pi}^0 = \{\pi_i^0\}_{i=1}^d$ and $\boldsymbol{\nu}^0 = \{\nu_{i,j}^0\}$; Maximum iteration step T; Batch size n^s; Stopping threshold ϵ.
Output: Converged $\boldsymbol{\pi}^t$ and $\boldsymbol{\nu}^t$.

Initialization of variational moment: $\{\boldsymbol{m},\boldsymbol{\lambda},\mathbb{E}[\alpha_{i,1}],\mathbb{E}[\alpha_{i,2}],\mathbb{E}\ln|\alpha_{i,1}|,\mathbb{E}\ln|\alpha_{i,2}|\}_{i=1}^d$:
- Initialize $\boldsymbol{m}^0$ by linear regression between $\boldsymbol{Y}$ and $\boldsymbol{X}$, and let $\boldsymbol{\lambda}^0 = (\boldsymbol{m}^0 \odot \boldsymbol{m}^0)^{-1}$;
- Set $\tilde{\boldsymbol{\nu}}^0 = \boldsymbol{\nu}^0$ and compute $E[\alpha_{i,1}],E[\alpha_{i,2}],\mathbb{E}\ln|\alpha_{i,1}|,\mathbb{E}\ln|\alpha_{i,2}|$ by Equation (10) and (11);

for $1 \le t \le T$ **do**
  Random Sampling a data subset with size n^s;
  Update $\tilde{\nu}_{0,1}^t$ and $\tilde{\nu}_{0,2}^t$ by $f_{\nu_{0,1}^{t-1}}(n^s)$ and $f_{\nu_{0,2}^{t-1}}(\boldsymbol{m}^{t-1},\boldsymbol{\lambda}^{t-1})$;
  for $1 \le i \le d$ **do**
    Update each $\tilde{\pi}_i^t$ by $f_{\pi_i^{t-1}}(\boldsymbol{m}^{t-1},\boldsymbol{\lambda}^{t-1},\tilde{\boldsymbol{\nu}}^{t-1})$;
    Update each $\tilde{\nu}_{i,1}^t$, $\tilde{\nu}_{i,2}^t$, $\tilde{\nu}_{i,3}^t$, $\tilde{\nu}_{i,4}^t$ by $f_{\nu_{i,1}^{t-1}}(\tilde{\boldsymbol{\pi}}^t)$, $f_{\nu_{i,2}^{t-1}}(\boldsymbol{m}^{t-1}, \boldsymbol{\lambda}^{t-1}, \tilde{\boldsymbol{\pi}}^{t-1})$, $f_{\nu_{i,3}^{t-1}}(\tilde{\boldsymbol{\pi}}^t)$, $f_{\nu_{i,4}^{t-1}}(\boldsymbol{m}^{t-1},\boldsymbol{\lambda}^{t-1},\tilde{\boldsymbol{\pi}}^t)$;
    Update m_i^t and λ_i^t by $f_m(\tilde{\pi}_i^t,\boldsymbol{m}^{t-1},\tilde{\boldsymbol{\nu}}^t)$ and $f_\lambda(\tilde{\boldsymbol{\nu}}^t)$;
  end for
  Update $\boldsymbol{\pi}^t = \tilde{\boldsymbol{\pi}}^t, \boldsymbol{\nu}^t = \tilde{\boldsymbol{\nu}}^t$;
  if $t \ge 3$ **then**
    $\boldsymbol{\pi}^{mean} = (\boldsymbol{\pi}^{t-2} + \boldsymbol{\pi}^{t-1} + \boldsymbol{\pi}^t)/3$;
    Early Stop if $|\boldsymbol{\pi}^t - \boldsymbol{\pi}^{mean}| < \epsilon$;
  end if
end for

C.5 Theoretical Result

It has been shown that our method, as well as others in Bayesian variable selection, has potentially strong selection consistency [48, 17, 84, 87]. Consider the following model with inverse Gamma prior:

$$\begin{aligned} y_n \,|\, (\phi(x_n), \mathbf{w}, \sigma^2) &\sim \mathcal{N}(\mathbf{w}\phi(x_n), \sigma^2 I), \\ \mathrm{w}_i \,|\, (\sigma^2, \mathrm{z}_i = 0) &\sim \mathcal{N}(0, \sigma^2 \tau_{0,N}^2), \\ \mathrm{w}_i \,|\, (\sigma^2, \mathrm{z}_i = 1) &\sim \mathcal{N}(0, \sigma^2 \tau_{1,N}^2), \\ p(\mathrm{z}_i = 1) = 1 - p(\mathrm{z}_i = 0) &= q_N, \\ \sigma^2 &\sim \mathrm{IG}(\alpha_1, \alpha_2), \end{aligned} \tag{12}$$

where i runs from 1 to d, $q_N, \tau_{0,N}, \tau_{1,N}$ are constants that depend on sample size N, and IG (α_1, α_2) is the Inverse Gamma distribution with shape parameter α_1 and scale parameter α_2. Under regular conditions (See conditions 4.1–4.5 in [57]), selection consistency is established:

Theorem 1. *Assume regular conditions hold, under the model with inverse Gamma prior, we have* $p(\mathrm{z} = t \,|\, \boldsymbol{Y}, \sigma^2) \xrightarrow{\mathrm{p}} 1$ *as* $n \to \infty$*, that is, the posterior probability of the true model goes to* 1 *as the sample size increases to* ∞*.*

More related works on Bayesian feature selection can be found in [26, 56].

C.6 Simulation Study

In this section, we will conduct a series of simulations to verify selection performance on an $i.i.d.$ dataset with varying sizes and dimensions. Here, we consider cases in the standard multivariate regression. We first generate each input predictor from a standard normal distribution: $x_{ni} \sim N(0,1)$ for $i = 1,...,d$, and thus we generate response variables by subsequently sampling $\beta_j \sim \mathrm{Uniform}(1,3)$ for $j = 1,...,k < d$ and $y_n \sim N(\sum_{i=1}^k \beta_i x_{ni}, 1)$. We then vary the values of d and k to find the potential influence in terms of True Positive Rate (TPR) and False Positive Rate (FPR). The results are shown in Table 9.

We repeat each case 50 times and present the mean and variance of TPR and FPR. The hyper-parameter setting is listed in Table 8. We vary n^s to study the influence of batch size. Overall, our method

illustrates the experimental selection consistency. When $n > d$, our method almost always selects the correct k variables with TPR close to 100% and successfully screens all unnecessary variables with FPR equal to 0%. Even under the less informative circumstance when n has equal or less amount with d, our method can still achieve great selection results with TPR above 90%. As n goes up, there is a uniform improvement in all cases in terms of TPR and FPR.

Table 8: Hyper-parameters setting in feature selection.

π_i	$\nu_{0,1}$	$\nu_{0,2}$	$\nu_{i,1}$	$\nu_{i,2}$	$\nu_{i,3}$	$\nu_{i,4}$	T	n^s	ϵ
0.5	1	1	1	1	5	1	1000	256	0.5

Table 9: Feature selection in terms of TPR/FPR.

d=100	k	n	n^s	TPR	FPR
Case 1	50	200	64	99.92%±0.39%	0.00%±0.00%
Case 2	50	200	128	99.92% ± 0.39%	0.00%±0.00%
Case 3	50	400	64	100.00% ± 0.00%	0.00%±0.00%
Case 4	50	400	128	100.00% ± 0.00%	0.00%±0.00%
Case 5	90	200	64	99.86%±0.42%	0.00%±0.00%
Case 6	90	200	128	99.93% ± 0.26%	0.00% ± 0.00%
Case 7	90	400	64	100.00% ± 0.00%	0.00% ± 0.00%
Case 8	90	400	128	100.00% ± 0.00%	0.00% ± 0.00%
d=300	k	n	n^s	TPR	FPR
Case 1	100	300	64	95.21%±2.22%	2.16%±1.52%
Case 2	100	300	256	96.46% ± 2.12%	2.31% ± 2.10%
Case 3	100	500	64	99.92% ± 0.27%	0.00% ± 0.00%
Case 4	100	500	256	100.00% ± 0.00%	0.00% ± 0.00%
Case 5	250	300	64	91.34%±2.92%	11.92%±6.79%
Case 6	250	300	256	91.95% ± 2.40%	14.56% ± 8.35%
Case 7	250	500	64	99.92% ± 0.17%	0.00% ± 0.00%
Case 8	250	500	256	99.92% ± 0.05%	0.00% ± 0.00%
d=500	k	n	n^s	TPR	FPR
Case 1	100	450	64	92.70%±2.56%	4.41%±1.67%
Case 2	100	450	256	92.89% ± 2.69%	4.90% ± 1.82%
Case 3	100	800	64	99.94% ± 0.23%	0.00% ± 0.00%
Case 4	100	800	512	100.00% ± 0.00%	0.00% ± 0.00%
Case 5	450	500	64	90.21%±2.56%	12.68%±6.38%
Case 6	450	500	256	92.06% ± 1.84%	16.04% ± 6.69%
Case 7	450	800	64	99.92% ± 0.13%	0.00% ± 0.00%
Case 8	450	800	512	100.00% ± 0.00%	0.00% ± 0.00%