Abstract

Deep learning models typically require massive amounts of annotated data to train a strong model for a task of interest. However, data annotation is time-consuming and costly. How to use labeled data from a related but distinct domain, or just a few samples to train a satisfactory model are thus important. To this end, models should resist overfitting to the specifics of the training data in order to generalize well to new data. This paper proposes a novel Worst-case Feature Risk Minimization (WFRM) method that helps improve model generalization. Specifically, we tackle a minimax optimization problem in feature space at each training iteration. Given the input features, we seek the feature perturbation that maximizes the current training loss and then minimizes the training loss of the worst-case features. By incorporating our WFRM during training, we significantly improve model generalization under distributional shift – Domain Generalization (DG) and in the low-data regime – Few-shot Learning (FSL). We theoretically analyse WFRM and find the key reason why it works better than ERM – it induces an empirical risk-based semi-adaptive $L_2$ regularization of the classifier weights, enabling a better risk-complexity trade-off. We evaluate WFRM on two data-efficient learning tasks, including three standard DG benchmarks, PACS, VLCS and OfficeHome and the most challenging FSL benchmark Meta-Dataset. Despite the simplicity, our method consistently improves various DG and FSL methods, leading to the new state-of-the-art performances in all settings. Codes & models will be released at https://github.com/jslei/WFRM.
1 Introduction

Deep learning models are data-hungry and require massive amounts of annotation to train strong models for tasks of interest. Unfortunately, high-quality task- and domain-specific annotated data is generally scarce. This leads researchers to attempt to train models on small training sets, or on different data distributions that will be used for deployment. However, both of these scenarios are subject to high risk of overfitting and poor generalization. These two data-efficient learning problems have been studied under the umbrella of Domain Generalization (DG) and Few-Shot Learning (FSL).

The study of DG emerged a decade ago (Blanchard et al., 2011; Muandet et al., 2013) and explores different ways to address distribution shift between train and test data. DG methods take a variety of approaches to learn domain-invariant features including kernel methods (Muandet et al., 2013), auto encoders (Ghifary et al., 2015; Li et al., 2018b) or parametric models (Li et al., 2017). Recently, with the renaissance of meta-learning methods (Finn et al., 2017; Ravi & Larochelle, 2017), these methods have been utilized for DG by meta-optimizing the model initialization (Li et al., 2018a), classifier regularizers (Balaji et al., 2018), and metric functions (Dou et al., 2019) for domain invariance. On the other hand, the FSL setting requires generalising knowledge from a set of known tasks (e.g., categories to recognize) to a novel target task given only a few labelled examples of the target task/category (Wang et al., 2020c). Meta learning is the primary method in the field of FSL and dedicated to offering a migratable pattern such as metric function (Snell et al., 2017; Sung et al., 2018; Vinyals et al., 2016), shared parameter initialization (Finn et al., 2017; Nichol et al., 2018), universal normalization strategy (Du et al., 2020) and generic optimization algorithm (Ravi & Larochelle, 2017). With the advancement of research, some unconventional methodologies, including data-augmentation (Li et al., 2020), adapter-based approaches (Xu et al., 2022) and foundation model based transfer learning (Hu et al., 2022) are proposed, leading to excellent outcomes. A key challenge for practical FSL is that, similar to the DG problem, there may also be distribution shift together with category shift between the source and target tasks in FSL (Triantafillou et al., 2019).

Unlike the popular methodological paradigms to achieve DG and FSL separately, we take the perspective of achieving both by using a unified minimax optimization for a given model. It’s known that adversarial training in image (Shankar et al., 2018), weight (Foret et al., 2020) or data distribution space (Sagawa et al., 2019) is beneficial for the model generalization, we follow this wisdom and explore it in feature space. In particular, we introduce the idea of minimizing the risk of worst-case features, and present a novel method of minimax-based feature risk minimization which can be applied to improving both DG and FSL problems.

Figure 1: A schematic illustration of our method. For each feature derived from authentic instance, WFRM utilizes the Prop. 1 to generate the corresponding worst-case feature which lies in its $\rho-$neighborhood and maximizes the training loss. Then, the loss from original and worst-case features are optimized simultaneously.

As illustrated in Figure 1 we propose a novel Worst-case Feature Risk Minimization (WFRM) method that facilitates the generalization of DG and FSL models. Our WFRM is formulated as a minmax optimization problem in feature space. It finds the perturbation of input features that maximizes the loss of current iteration, and then minimizes the loss from the perturbed worst-case features. We derive an analytic solution to this loss maximization problem under
mild conditions. We show that our WFRM can be efficiently implemented in closed form without expensive iterative optimization. Furthermore, we theoretically analyze the mechanism of our WFRM, and find that minimizing the loss of the worst-case features implicitly provides an $L_2$ regularization (a.k.a weight decay) of the classifier’s weights during optimization. Different to normal $L_2$ regularization which is separate from the training loss, the introduced WFRM $L_2$ regularization is associated with the training loss, resulting in that the penalty will disappear when the training loss becomes small enough. Therefore, this is a semi-adaptive $L_2$ regularization controlled by the empirical risk of the training model and the perturbation radius. The efficacy of this adaptive regulariser is supported by the recent paper (Li et al., 2022a), that shows how model complexity plays an important role of domain generalization performance.

Our WFRM is beneficial for various cross-domain tasks including both DG and FSL thanks to its ability to find an improved risk-complexity trade-off. It is a simple plug-and-play module that can be applied to improve optimization of any base DG method, such as ERM, EISNet (Wang et al., 2020b), DDAIG (Zhou et al., 2020a) and any FSL method, such as eTT (Xu et al., 2022). We evaluate our method on three different DG benchmarks, PACS, VLCS and OfficeHome, and the largest-scale FSL benchmark Meta-Dataset, and consistently demonstrate its new state-of-the-art performance in all settings.

Contributions. Our main contributions are: (1) We propose a novel worst-feature risk minimization method that facilitates generalization against distribution shift. (2) We give the analytic solution of our WFRM objective under mild conditions, and show that the inner WFRM objective can be solved efficiently in closed form. (3) We further present theoretical analysis showing that our WFRM provides a risk-based semi-adaptive $L_2$ regularization for the classifier. (4) Extensive experiments on benchmarks of DG and FSL show the broad applicability and state of the art performance of our method.

2 Related Work

Domain Generalization Various domain generalization methods have been proposed in the literature. Many focus on learning the domain-invariant feature representations, through learning kernel-based subspaces (Muandet et al., 2013), learning invariant embeddings by cross-domain auto reconstruction (Ghifary et al., 2015), metric learning (Motiian et al., 2017), variance regularizer (Krueger et al., 2021), or model disentanglement (Khosla et al., 2012; Li et al., 2017). Inspired by the meta-learning methods (Finn et al., 2017), researchers have also studied meta-learning based DG methods, including different meta-learning optimization strategies of model initialization (Li et al., 2018a), classifier regulariser (Balaji et al., 2018), metric learning (Dou et al., 2019), and loss learning (Gao et al., 2022). Data augmentation-based DG methods are also popular, including strategies of augmenting the training data in the pixel (Zhou et al., 2020a,b, 2020c), Shankar et al. (2018), frequency (Xu et al., 2021) or feature (Li et al., 2021) spaces. Differently to these methods, our work proposes a novel worst-case risk minimization method, with theoretical analysis to show that it corresponds to a novel adaptive regularization loss. Some recent theoretical analysis of DG has shown that model-complexity control is important for DG (Li et al., 2022a). This is in line with our contribution, for which our own theoretical analysis shows that our method provides semi-adaptive control of model complexity.

Few-shot Learning aims to transfer prior knowledge learned from known tasks to help learn novel tasks with limited data. Meta-learning and transfer learning are the two main approaches to solving FSL problems. Meta-learning methods typically provide a good model initialization (Finn et al., 2017) or metric (Snell et al., 2017) or learning novel tasks in the way of using a few samples. In contrast, other studies focused on transfer learning (Liu et al., 2020, Gidaris et al., 2019, Afrasiyabi et al., 2020, Chen et al., 2019) and fine-tuning (Li et al., 2022a, Xu et al., 2022) to improve few-shot generalization. Differently, we propose a new worst-case risk minimization loss that is beneficial for mitigating the effect of the domain gap in FSL.

Adversarial Attack Adversarial attack (Goodfellow et al., 2014) or black-box (Chen et al., 2017) or black-box (Goodfellow et al., 2014) or black-box (Chen et al., 2017) is a topical area that studies how worst-case perturbations in image space can deceive a trained machine learning model. Existing attack techniques can be categorised as gradient based (Goodfellow et al., 2014) or black-box (Chen et al., 2017) or black-box (Chen et al., 2017). Our proposed WFRM is relevant to gradient based adversarial attacks, and few-shot generalization. Differently, we propose a new worst-case risk minimization loss that is beneficial for mitigating the effect of the domain gap in FSL.
Ideas related to the adversarial attack and min-max optimization during learning have been used in training adversarially robust models (Madry et al., 2018), training models with improved in-distribution generalization by sharpness-aware minimization (Foret et al., 2020), improving models robustness against group shift (Sagawa et al., 2019), and training models robust to parameter corruption (Sun et al., 2021). While a few papers have studied adversarial learning for DG (Shankar et al., 2018; Sagawa et al., 2019) and FSL (Goldblum et al., 2020; Wang & Deng, 2021), we propose a feature-space adversarial learning to address them two jointly, and a theoretical analysis of why it is helpful. Finally, we remark that existing adversarial attacks require iterative optimization to find effective perturbations (Madry et al., 2018), and are thus slow to use during training. Our particular WFRM formulation enables a closed-form solution for efficiently finding the worst-case perturbation for fast training of robust models.

3 Methodology

We denote the input and ground truth data pairs as \((x, y)\), and the model we want to train as \(g_W \circ f_{\Theta}(x)\), where \(f_{\Theta}\) is the nonlinear feature extractor parameterized by \(\Theta\), such as CNNs and transformers; and \(g_W\) is a classifier with parameters \(W\). We concern with the following two types of data-efficient learning.

Domain Generalization. We have the training set of \(S\) source domains \(D = \{D_1, \ldots, D_S\}\), where \(D_i = \{(x_j, y_j)\}_{j=1}^{N_i}\) containing \(N_i\) paired data and labels. The goal of DG is to train a model on the source domains, prior to testing it on an unseen target domain \(D_{S+1}\), by minimizing the empirical risk as

\[
\min_{\Theta, W} \mathcal{L}_{ERM} = \frac{1}{S} \sum_{i=1}^{S} \frac{1}{N_i} \sum_{j=1}^{N_i} \ell(g_W \circ f_{\Theta}(x_j), y_j),
\]

(1)

where \(\ell\) is the cross-entropy loss in our task. The data and label distributions of unseen target domain are normally different from those of source domains.

Few-shot Learning. We further formulate few-shot learning in the meta-learning paradigm. Generally, there are two sets of data: meta-train set \(D_s = \{(x_i, y_i), y_i \in C_s\}\) and meta-test set \(D_t = \{(x_j, y_j), y_j \in C_t\}\) containing the data from source dataset \(C_s\) and target dataset \(C_t\), respectively \((C_s \cap C_t = \emptyset)\). These data are possibly collected from different domains; and FSL trains a model on \(D_s\) that benefits subsequent generalization \(D_t\) (for example by transferring an initial set of parameters \(\Theta\)). We focus on the meta-testing or adaptation phase of learning on the novel data, where there are few samples from each category of \(C_t\), \(D_t = \{(x_j, y_j)\}_{j=1}^{K}\). Here \(K\) can often be a number \(\leq 20\) (Xu et al., 2022). Thus FSL optimizes the loss as

\[
\min_{\Theta, W} \mathcal{L}_{ERM} = \frac{1}{K} \sum_{j=1}^{K} \ell(g_W \circ f_{\Theta}(x_j), y_j),
\]

(2)

where \(g_W\) can be a linear classifier, or estimated categorical centroids for prototypical methods. Different to DG, \(f_{\Theta}\) is typically a pre-trained model from the known tasks. Note that we simplify the subscript \(\Theta\) and \(W\) in next sections.

3.1 Worst-case Feature Risk Minimization

Beyond classical empirical risk minimization in Eq. [1] and Eq. [2] we develop the key contribution of a minimax optimization for Worst-case Feature Risk Minimization (WFRM). Our key idea is to introduce a feature perturbation that leads to the maximal loss value in the neighborhood of sample feature. It can thus be formulated as the worse-case feature risk minimization by minimizing the training loss of the perturbed features.

To maintain the denseness of the perturbed feature space, we use \(L_2\) ball to find the perturbation. The overall optimization on a dataset containing \(N\) data pairs is formulated as follows,

\[
\min_{\Theta, W} \mathcal{L}_{pert} = \frac{1}{N} \sum_{i=1}^{N} \max_{\|\epsilon_i\|_2 \leq \rho} \ell(g_W(f_{\Theta}(x_i) + \epsilon_i), y_i)
\]

(3)

where \(\epsilon\) is the feature perturbation and \(\rho\) is the upper bound of its \(L_2\) norm.
Thus the total training loss is summed as
\[ L_{\text{total}} = (1 - \alpha) \cdot L_{\text{ERM}} + \alpha \cdot L_{\text{pert}}, \]
where \( \alpha \) is the weighting coefficient. We then give the theoretical insights and implementation details of \( L_{\text{pert}} \) with the ERM model.

### 3.1.1 Analytic Solver of WFRM

We give the details of solving the maximization in the inner loop of Eq. 3 under the mild conditions.

**Proposition 1.** Suppose the loss function is a convex function of the features and the gradient of loss function with respect to feature \( \nabla_h \ell \) is \( L \)-Lipschitz continuous, where \( h = f(x) \). Then for any given feature \( h \), the maximum value of the loss function in the \( \rho \)-neighbourhood of \( h \) can be bounded by the value of loss function at \( h + \epsilon \). Namely,
\[
\ell(h + \epsilon) \leq \max_{||\epsilon||_2 \leq \rho} \ell(h + \epsilon) = \ell(h + \epsilon^*) + \frac{L}{2} \rho^2.
\]

The proof is in the appendix. Using the result in proposition 1, now \( L_{\text{pert}} \) can be efficiently minimized by optimizing \( \ell(h + \epsilon) \) as \( \rho \) is a hyperparameter and \( L \) is a constant. Note that our conditions are actually mild, in the sense that if the classifier is a linear layer, the cross-entropy loss is thus a convex function of the features extracted from the penultimate layer of networks. The combination of linear classifier and cross-entropy loss is widely used in recognition tasks. So WFRM can be easily applied in most scenarios. Empirically, we also tried other variants (e.g., iterative schemes such as PGD (Madry et al., 2018)) to deal with the minimax problem but achieved little gain over WFRM, despite being much slower. As a numeric method for a constrained convex minimization problem, projected gradient descent has difficulty in solving the constrained convex maximization problem, which is equivalent to the constrained concave minimization problem. But by using the derived upper bound, the predicament can be addressed easily.

### 3.2 Theoretical Analysis of WFRM

In this part, we conduct a brief analysis on the logistic model and point out the key ingredient of why our method works.

**Proposition 2.** Let us assume the labels \( y \in \{-1, 1\} \). The loss function of the logistic regression which incorporates WFRM is \( \hat{L}_{\text{pert}} = \log(1 + e^{\rho||w||_2 - y(w^T x + b)}) \).

As shown in Proposition 2, whose proof is given in appendix, \( \hat{L}_{\text{pert}} \) will additionally contain terms with respect to the weight vector \( w \). The gradient of \( \hat{L}_{\text{pert}} \) with respect to \( w \) is shown as follows:
\[
\nabla_w \hat{L}_{\text{pert}} = \frac{\rho w / ||w||_2 - y x}{1 + e^{y(w^T x + b) - \rho ||w||_2}} \\
\triangleq \beta(w)(\rho w / ||w||_2 - y x) \\
\triangleq r(w) - \beta(w)y x.
\]

Compared with the gradient of loss produced by the raw features which is \(-y x / (1 + e^{y(w^T x + b)})\), the main difference is that \( \nabla_w \hat{L}_{\text{pert}} \) contains a term, i.e. \( r(w) \) in the same direction as \( w \). From this point of view, the effect produced by our method is similar to \( L_2 \) regularization. But there also exits a noticeable difference. For a given batch, when the current model fits data well, the training loss becomes small. This means that \( \beta(w) \) will be small. Note that the norm of \( r(w) \) is \( \rho \beta(w) \). Thus, in this case, the length of \( r(w) \) will be small. Therefore, even if the norm of the current weight vector is large, the force driving the weight vector close to the origin could still be moderate. On the contrary, if the current model has bad performance, \( r(w) \) will push the weight vector close to the origin to a greater extent. That means WFRM enables the regularization strength to be adaptively adjusted according to how well the current model fits the data. Additionally, since it is possible that the parameters that make the model perform well are far from the origin, compared to normal \( L_2 \) regularization, WFRM is able to tolerate weights with bigger scale, which enables optimization algorithms to search for parameters in a larger space. Hence, better parameters can be found.
In image recognition tasks such as DG and FSL, linear classifiers and cross-entropy loss are widely employed. These components share consistency with logistic regression. Additionally, as pointed in (Snell et al., 2017), ProtoNets adopted in eTT (Xu et al., 2022) can be also re-interpreted as a linear classifier. Therefore, the above analysis can be easily extended to these FSL ProtoNets. In the FSL experiments, we insert WFRM before the last linear projection layer, which has no activation function applied in its outputs so that its weights can also benefit from our regularization.

Discussion. We further discuss the relation and difference of our WFRM and recent adversarial training methods such as FGSM (Goodfellow et al., 2014), SAM (Foret et al., 2020) and GroupDRO (Sagawa et al., 2019). Particularly, (1) the adversarial attack method (Goodfellow et al., 2014), tries to synthesize adversarial examples via perturbing the raw image pixels. The augmented adversarial examples can also be utilized to train the model. Intuitively, it is easy to directly extend FGSM to attack features: we can easily derive its logistic regression formulation to minimize \( \tilde{L}_{FGSM} = \log(1 + e^{\rho ||w||_1 - y(w^T x + b)}) \) along the line of our Proposition 2. However, directly applying FGSM at feature level will lead to an \( L_1 \) regularization also seen in (Goodfellow et al., 2014) while WFRM is the \( L_2 \) version. \( L_1 \) is commonly used for model sparsity, with a different variant of controlling model complexity. Unfortunately, empirical results demonstrate that the performance of simply extending FGSM in feature space is inferior to our WFRM as shown in Table 6. Furthermore, (2) Our WFRM is relevant to SAM (Foret et al., 2020) as well, which conducts adversarial training in the weight space in classical supervised learning. Although the minimax optimization in WFRM is related to SAM (Foret et al., 2020), SAM conducts the minimax optimization on all learnable parameters in the network, and makes backward operation twice in each training iteration, which is notoriously slow. In contrast, our WFRM conducts the minimax optimization on the penultimate features only, which is more efficient and incurs a similar training cost as ERM as shown in Sec. 4.3. (3) Finally, the concept of robust optimization in the data distribution space has been explored by GroupDRO (Sagawa et al., 2019). However, it only considers the convex hull of the training distributions, which may not be sufficient for training a satisfactory model when the difference between the training and test domains is significant. In such cases, GroupDRO may not perform well. Our empirical results in Table 5 demonstrate that WFRM outperforms GroupDRO.

4 Experiments

4.1 Domain Generalization

To verify the effectiveness of our model, we conduct experiments on three DG benchmarks: PACS (Li et al., 2017) (4 domains, 9,991 images, 7 classes), VLCS (Fang et al., 2013) (4 domains, 10,729 images, 5 classes) and Office-Home (Venkateswara et al., 2017) (4 domains, 15,588 images, 65 classes). We use PyTorch (Paszke et al., 2019) and run our experiments on a GeForce GTX 1080 Ti GPU.

Baselines. We compare our method with the existing state of the art DG methods: ERM, the standard empirical risk minimizer, which is a strong baseline as pointed out in (Li et al., 2017; Gulrajani & Lopez-Paz, 2021). DANN (Ganin et al., 2016), a domain adaptation method repurposed for DG (Li et al., 2019). CCSA (Mottian et al., 2017), a metric-learning DG method. MAML (Finn et al., 2017), the meta-learning few-shot learning method repurposed for DG (Li et al., 2019). MLDG (Li et al., 2018a), a MAML inspired meta-learning DG method. CrossGrad (Shankar et al., 2018), a Bayesian network that augments the input data by maximizing the domain classification loss. MetaReg (Balaji et al., 2018), a DG method meta-training classifier regularizer. MMD-AAE (Li et al., 2018b), an adversarial auto encoder with domain-invariant feature. JiGen (Carlucci et al., 2019), a self-supervised learning DG method. Epi-FCR (Li et al., 2019), a first-order meta-learning DG method learned by simulated episodes. DDAIG (Zhou et al., 2020a), an image augmentation DG method by adversarial training. RSC (Huang et al., 2020), a DG method learns generalizable model by self-reinforcement. EISNet (Wang et al., 2020b), a metric learning based DG method. L2A-OT (Zhou et al., 2020b), a data augmentation method using optimal transport. SAM (Foret et al., 2020), a worst-case weight space perturbation, which is repurposed for DG by us. SFA-A (Li et al., 2021), a feature augmentation based DG method that augments the feature space by learned Gaussian noise. And our WFRM, which minimizes the risk of the worst-case feature.

4.1.1 Evaluation on PACS

Implementation details. We use ResNet-18 (ImageNet pretrained) as our backbone and follow the official train/val split as per (Li et al., 2017). The network is trained with M-SGD, batch size 16, momentum 0.9, learning rate 0.002
and weight decay 0.0005 for 50 epochs. No data augmentation strategy is used but all images are resized to $224 \times 224$. During training, our WFRM is inserted after the global average pooling layer. $\alpha$ is set to 0.5 throughout the experiments and $\rho$ is set to 1.5. Unless otherwise specified, we follow the train/val/test split protocol in (Wang et al., 2020a; Zhou et al., 2021) and utilize the validation set to determine the value of $\rho$ in all DG experiments.

### Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>A</th>
<th>C</th>
<th>P</th>
<th>S</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DANN</td>
<td>77.1</td>
<td>73.8</td>
<td>94.0</td>
<td>74.3</td>
<td>80.8</td>
</tr>
<tr>
<td>MAML</td>
<td>78.3</td>
<td>76.5</td>
<td>95.1</td>
<td>72.6</td>
<td>80.6</td>
</tr>
<tr>
<td>MLGD</td>
<td>79.5</td>
<td>77.3</td>
<td>94.3</td>
<td>71.5</td>
<td>80.7</td>
</tr>
<tr>
<td>CrossGrad</td>
<td>78.7</td>
<td>73.3</td>
<td>94.0</td>
<td>65.1</td>
<td>77.8</td>
</tr>
<tr>
<td>MetaReg</td>
<td>79.5</td>
<td>75.4</td>
<td>94.3</td>
<td>72.2</td>
<td>80.4</td>
</tr>
<tr>
<td>Epi-FCR</td>
<td>82.1</td>
<td>77.0</td>
<td>93.9</td>
<td>73.0</td>
<td>81.5</td>
</tr>
<tr>
<td>RSC</td>
<td>78.9</td>
<td>76.9</td>
<td>94.1</td>
<td>76.8</td>
<td>81.7</td>
</tr>
<tr>
<td>EISNet</td>
<td>81.9</td>
<td>76.4</td>
<td>95.9</td>
<td>74.3</td>
<td>82.2</td>
</tr>
<tr>
<td>L2A-OT</td>
<td>83.3</td>
<td>78.2</td>
<td>96.2</td>
<td>73.6</td>
<td>82.8</td>
</tr>
<tr>
<td>SFA-A</td>
<td>81.2</td>
<td>77.8</td>
<td>93.9</td>
<td>73.7</td>
<td>81.7</td>
</tr>
<tr>
<td>SAM</td>
<td>80.9</td>
<td>76.9</td>
<td>93.7</td>
<td>76.4</td>
<td>82.0</td>
</tr>
<tr>
<td>ERM</td>
<td>77.1</td>
<td>78.6</td>
<td>94.0</td>
<td>70.3</td>
<td>80.0</td>
</tr>
<tr>
<td>ERM+WFRM</td>
<td>80.4</td>
<td>77.5</td>
<td>94.5</td>
<td>75.9</td>
<td>82.1</td>
</tr>
<tr>
<td>DDAIG (*)</td>
<td>82.4</td>
<td>74.0</td>
<td>93.7</td>
<td>71.7</td>
<td>80.4</td>
</tr>
<tr>
<td>DDAIG+WFRM</td>
<td>84.1</td>
<td>78.0</td>
<td>92.3</td>
<td>72.5</td>
<td>81.7</td>
</tr>
<tr>
<td>EISNet (*)</td>
<td>82.5</td>
<td>75.8</td>
<td>96.2</td>
<td>74.7</td>
<td>82.3</td>
</tr>
<tr>
<td>EISNet+WFRM</td>
<td>83.7</td>
<td>77.4</td>
<td>95.6</td>
<td>77.6</td>
<td>83.6</td>
</tr>
</tbody>
</table>

### Table 1: Results on PACS with ResNet-18 (Top-1 accuracy, %). * reproduced using their codebase.

### Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>V</th>
<th>L</th>
<th>C</th>
<th>S</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIDDG</td>
<td>64.4</td>
<td>63.1</td>
<td>88.8</td>
<td>62.1</td>
<td>69.6</td>
</tr>
<tr>
<td>CCSA</td>
<td>67.1</td>
<td>62.1</td>
<td>92.3</td>
<td>59.1</td>
<td>70.2</td>
</tr>
<tr>
<td>DBAGD</td>
<td>70.0</td>
<td>63.5</td>
<td>93.6</td>
<td>61.3</td>
<td>72.1</td>
</tr>
<tr>
<td>MMD-AAE</td>
<td>67.7</td>
<td>62.6</td>
<td>94.4</td>
<td>64.4</td>
<td>72.3</td>
</tr>
<tr>
<td>MLGD</td>
<td>67.7</td>
<td>61.3</td>
<td>94.4</td>
<td>65.9</td>
<td>72.3</td>
</tr>
<tr>
<td>Epi-FCR</td>
<td>67.1</td>
<td>64.3</td>
<td>94.1</td>
<td>65.9</td>
<td>72.9</td>
</tr>
<tr>
<td>JiGen</td>
<td>70.6</td>
<td>60.9</td>
<td>96.9</td>
<td>64.3</td>
<td>73.2</td>
</tr>
<tr>
<td>MASF</td>
<td>69.1</td>
<td>64.9</td>
<td>94.8</td>
<td>67.6</td>
<td>74.1</td>
</tr>
<tr>
<td>EISNet</td>
<td>69.8</td>
<td>63.5</td>
<td>97.3</td>
<td>68.0</td>
<td>74.7</td>
</tr>
<tr>
<td>SFA-A</td>
<td>70.4</td>
<td>62.0</td>
<td>97.2</td>
<td>66.2</td>
<td>74.0</td>
</tr>
<tr>
<td>SAM</td>
<td>68.5</td>
<td>57.8</td>
<td>99.5</td>
<td>64.8</td>
<td>72.7</td>
</tr>
<tr>
<td>ERM</td>
<td>68.3</td>
<td>60.6</td>
<td>97.6</td>
<td>63.5</td>
<td>72.5</td>
</tr>
<tr>
<td>ERM+WFRM</td>
<td>72.0</td>
<td>60.6</td>
<td>97.9</td>
<td>67.8</td>
<td>74.6</td>
</tr>
<tr>
<td>DDAIG (*)</td>
<td>64.4</td>
<td>59.4</td>
<td>95.8</td>
<td>63.1</td>
<td>70.6</td>
</tr>
<tr>
<td>DDAIG+WFRM</td>
<td>65.8</td>
<td>60.0</td>
<td>96.0</td>
<td>61.8</td>
<td>70.9</td>
</tr>
<tr>
<td>EISNet (*)</td>
<td>72.2</td>
<td>61.5</td>
<td>98.1</td>
<td>63.3</td>
<td>73.8</td>
</tr>
<tr>
<td>EISNet+WFRM</td>
<td>73.5</td>
<td>62.9</td>
<td>97.9</td>
<td>65.0</td>
<td>74.8</td>
</tr>
</tbody>
</table>

### Table 2: Results on VLCS with AlexNet (Top-1 accuracy, %). * reproduced using their codebase.

### Results

Table 1 summarizes the results on PACS. Our WFRM achieves comparable results to state-of-the-art when built over ERM and a clear improvement margin of 2.1% over the vanilla ERM method. Our WFRM is a plug-and-play module and we also apply our WFRM over two prior arts DDAIG and EISNet. We can see they are all improved by our module with accuracy margins of 1.3%, leading to the new best result on this benchmark. From the results here, we can see SAM (Foret et al., 2020) performs comparably well as our WFRM, demonstrating adversarial training in weight and feature space can both improve DG performance but our WFRM is more efficient as we will show. And our WFRM works more effectively on improving DG performance when compared with the Gaussian noise feature space perturbation method SFA-A (Li et al., 2021).

### 4.1.2 Evaluation on VLCS

#### Implementation details

We use AlexNet (Krizhevsky et al., 2017) (ImageNet pretrained) as our backbone and follow the train/val protocols as per (Wang et al., 2020b). The network is trained with M-SGD, batch size 64, momentum 0.9, learning rate 0.0002 and weight decay 0.0005 for 30 epochs. Following (Wang et al., 2020b), we use random resized cropping, horizontal flipping and color jittering for data augmentation. During training, our WFRM is inserted after the FC7 layer and $\rho$ is set to 6.5.

#### Results

Table 2 shows the overall results, demonstrating the state-of-the-art performance of our WFRM. Our method outperforms all the other previous state-of-the-art method in terms of the average accuracy and achieves the best performance on the target domain PASCAL. Again, our WFRM gains accuracy boost over ERM, DDAIG and EISNet by 2.1%, 0.3% and 1.0% respectively, showing the effectiveness of our method. Both the SFA-A (Li et al., 2021) and our WFRM work well in this case while our WFRM gains slightly better results. More interestingly, our WFRM outperforms SAM (Foret et al., 2020) the weight space adversarial training method with a clear margin of 1.9%.

### 4.1.3 Evaluation on OfficeHome

#### Implementation details

We use ResNet-18 (ImageNet pretrained) as the backbone model and follow the train/val split in (Zhou et al., 2021). The network is trained with M-SGD, batch size 32, momentum 0.9, learning rate 0.001 and weight decay 0.0005 for 50 epochs. The learning rate is decayed by 0.1 at the 40th epoch. Our data augmentation
strategy includes random resized cropping, horizontal flipping and color jittering. During training, WFRM is again inserted after the global average pooling layer and $\rho$ is set to 0.45.

**Results.** The main results are shown in Table 3. Due to the nature of the benchmark, it seems all the existing DG methods hardly gain noticeable improvement over ERM and mostly are even worse. However, it is found that our WFRM, despite its simplicity, still improves the vanilla, DDAIG and EISNet by clear margins 0.8%, 1.0% and 0.5% on average and achieves the new state-of-the-art performance. In this case, SAM (Foret et al., 2020) also works well, demonstrating the adversarial training in weight space is also helpful.

**4.2 Few-shot Learning**

To verify the efficacy of our method on few-shot learning as well, we evaluate WFRM on Meta-Dataset (Triantafillou et al., 2019), which is currently the most challenging FSL benchmark and contains 10 diverse sub-datasets.

**Baselines.** Following (Xu et al., 2022), several state-of-the-art FSL methods are chosen for fair comparisons. Prototypical network (Snell et al., 2017), a metric-based method which classifies query samples by finding their nearest class prototype estimated from the support samples. CTX (Doersch et al., 2020), a protonet-inspired metric learning method which learns query-specific categorical centroids using cross-attention mechanism. TSA (Li et al., 2022b), an adapter-based approach. eTT (Xu et al., 2022), a recent adapter-based few-shot learning method built over vision transformer (Dosovitskiy et al., 2021). P>F>M (Hu et al., 2022), a pretrain-meta-train-finetuning few-shot learning pipeline. And we further include more methods to verify the effectiveness of WFRM. FGSM-F (Goodfellow et al., 2014), the variant of FGSM which we repurposed to conduct adversarial training on eTT (Xu et al., 2022) in the feature space. SUPMoCo (Majumder et al., 2021), a supervised contrastive learning based on MoCo (He et al., 2020). Deta (Zhang et al., 2023), a test-time adaptation method by filtering out noisy information.

**Implementation.** Following the experimental setup of eTT, we use DINO ViT-small (Caron et al., 2021) pretrained on the meta-train split of ImageNet (Deng et al., 2009). We plug WFRM before the final linear transformation layer and fine-tune eTT with our WFRM on the meta-test splits of all the 10 sub-datasets for evaluation. The competitors and hyperparameters (other than $\rho$ and $\alpha$) are exactly the same as (Xu et al., 2022). For our algorithm-specific hyperparameters, we use the selection method in (Xu et al., 2022). And $\rho$ and $\alpha$ are respectively set to 10 and 0.5 for all 10 datasets. Our experiments are run on four Tesla V100 GPUs.
Figure 2: The two leftmost figures show the sensitivity analysis of different choices of $\rho$ on VLCS with AlexNet and on Meta-Dataset with eTT, respectively. And the two rightmost figures show the sensitivity analysis of different choices of $\alpha$. The red dotted line is the accuracy of ERM or eTT.

![Graph](image1)

![Graph](image2)

![Graph](image3)

![Graph](image4)

**Table 5: Results on DomainBed benchmark.**

<table>
<thead>
<tr>
<th>Methods</th>
<th>PACS</th>
<th>VLCS</th>
<th>OfficeHome</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>80.5 ± 0.7</td>
<td>75.6 ± 0.8</td>
<td>60.8 ± 0.5</td>
<td>72.3</td>
</tr>
<tr>
<td>RSC (Huang et al. 2020)</td>
<td>80.1 ± 0.4</td>
<td>75.4 ± 0.4</td>
<td>58.4 ± 0.2</td>
<td>71.3</td>
</tr>
<tr>
<td>DANN (Hu et al. 2020)</td>
<td>80.3 ± 0.7</td>
<td>76.0 ± 0.0</td>
<td>59.8 ± 0.2</td>
<td>72.0</td>
</tr>
<tr>
<td>IIMT (Yan et al. 2020)</td>
<td>80.0 ± 0.5</td>
<td>75.0 ± 0.5</td>
<td>60.1 ± 0.7</td>
<td>71.7</td>
</tr>
<tr>
<td>GroupDRO (Sagawa et al. 2019)</td>
<td>81.3 ± 0.6</td>
<td>74.6 ± 0.2</td>
<td>59.9 ± 0.5</td>
<td>71.9</td>
</tr>
<tr>
<td>VREx (Krueger et al. 2021)</td>
<td><strong>81.7 ± 0.4</strong></td>
<td>74.7 ± 0.9</td>
<td>59.1 ± 0.2</td>
<td>71.8</td>
</tr>
<tr>
<td>CAD (Dubois et al. 2021)</td>
<td>81.1 ± 0.3</td>
<td>75.4 ± 0.1</td>
<td>60.3 ± 0.3</td>
<td>72.3</td>
</tr>
<tr>
<td>CausIRL (Chevalley et al. 2022)</td>
<td>81.0 ± 0.1</td>
<td>76.0 ± 0.5</td>
<td>60.8 ± 0.4</td>
<td>72.6</td>
</tr>
<tr>
<td>Transfer (Zhang et al. 2021)</td>
<td>81.2 ± 0.4</td>
<td>73.1 ± 0.8</td>
<td>57.8 ± 0.3</td>
<td>70.7</td>
</tr>
<tr>
<td>WFRM</td>
<td>79.2 ± 0.4</td>
<td><strong>77.5 ± 0.2</strong></td>
<td><strong>61.8 ± 0.4</strong></td>
<td><strong>72.8</strong></td>
</tr>
</tbody>
</table>

**Results.** The results of randomly sampled 600 episodes for each dataset are shown in Table 4. And the 95% confidence interval can be found in the appendix. Our WFRM improves eTT (Xu et al., 2022) by around 3% on average on 10 datasets and achieves the best on six out of the ten datasets, which remarkably shows that our method effectively improves model generalization under the low-data regime even with source knowledge bias embedded in the model. Furthermore, FGSM-F, the repurposed adversarial learning in the feature space, also leads to a performance improvement over eTT (Xu et al., 2022) yet still underperforms our WFRM by about 1.1% on average.

**4.3 Further Analysis**

**Sensitivity of hyperparameters.** We conduct sensitivity analysis on the hyperparameters introduced in our method and show the results of varying $\rho$ and $\alpha$ in Figure 2. It is clear that our method is not strongly sensitive to the different choices of $\rho$ both in DG and FSL tasks, outperforming the baseline consistently. Moreover, the model incorporating WFRM is able to achieve performance improvement despite the different value of $\alpha$. In fact, $\alpha$ is set to 0.5 in our all experiments, which means WFRM just introduces one additional hyperparameter to some extent.

**Results on DomainBed.** To verify the efficacy of our WFRM more thoroughly, we further conduct experiments on DomainBed benchmarks. We choose ResNet-18 rather than ResNet-50 in DomainBed as our backbone for efficiency while following the wisdom pointed out in (Huang et al., 2022; Ye et al., 2022) that a smaller base model could provide more effective testbed for generalization ability. Then, we follow exactly the default settings with the training-domain validation set used for model selection. Besides the base hyperparameters, we include our algorithm-specific hyperparameter $\rho$, which is drawn from random.choice([0, 1]). Some well-known DG methods RSC (Huang et al., 2020), GroupDRO (Sagawa et al., 2019), DANN (Hu et al., 2020), CAD (Dubois et al., 2021), CausIRL (Chevalley et al., 2022), IIMT (Yan et al., 2020), Transfer (Zhang et al., 2021) and VREx (Krueger et al., 2021) are also implemented for a fair comparison. The overall results are shown in Table 5 and more detailed results are presented in appendix. We can see that our method outperforms the DomainBed discovered strong baseline ERM by a clear margin of 1.9% on VLCS, 1.0% on OfficeHome and comparable performance to ERM and several DG methods on PACS, further
Figure 3: (a) Ablation study of different choices of weight decay on VLCS. The red and green dotted line are the accuracy values of ERM and WFRM, respectively. We tuned the weight decay coefficients of the last layer from $1e^{-5}$ to $5e^{-4}$. The accuracy peaks when $wd = 1.0$. For WFRM, the normal weight decay is disabled. (b) The plotted curves of the Frobenius norm of final classifier weights (dotted line) and empirical loss (solid line) along training iterations on VLCS(V). (c) Inverse Visualization of the worst-case features. WFRM is able to modify the image style while maintaining the feature semantics.

indicating the efficacy of our proposed method. Moreover, compared with ERM, our WFRM has a smaller standard deviation overall, which indicates our WFRM can improve the training stability of ERM.

**Comparison with more adversarial attacks.** Our WFRM is relevant to adversarial attacks (Goodfellow et al., 2014; Madry et al., 2018). We also compare our method with the model trained with the adversarial attacks in more situations. From the results in Table 6 we can see that FGSM-based adversarial training can also benefit DG performance. Vanilla FGSM and PGD were proposed for the pixel perturbation. Thus, we also implement a variant of the feature-level attack but find they still underperform our WFRM by a large margin.

<table>
<thead>
<tr>
<th>Methods</th>
<th>V</th>
<th>L</th>
<th>C</th>
<th>S</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>68.3</td>
<td>60.6</td>
<td>97.6</td>
<td>63.5</td>
<td>72.5</td>
</tr>
<tr>
<td>image-level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FGSM</td>
<td>67.9</td>
<td>60.1</td>
<td>98.3</td>
<td>65.9</td>
<td>73.1</td>
</tr>
<tr>
<td>PGD</td>
<td>68.8</td>
<td>60.0</td>
<td>97.6</td>
<td>64.8</td>
<td>72.3</td>
</tr>
<tr>
<td>WFRM</td>
<td>69.2</td>
<td>60.4</td>
<td>97.9</td>
<td>65.2</td>
<td>73.2</td>
</tr>
<tr>
<td>feature-level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FGSM</td>
<td>71.5</td>
<td>60.9</td>
<td>96.9</td>
<td>64.9</td>
<td>73.5</td>
</tr>
<tr>
<td>PGD</td>
<td>69.2</td>
<td>56.2</td>
<td>94.6</td>
<td>65.5</td>
<td>71.4</td>
</tr>
<tr>
<td>WFRM</td>
<td>72.0</td>
<td>60.6</td>
<td>97.9</td>
<td>67.8</td>
<td>74.6</td>
</tr>
</tbody>
</table>

Table 6: Comparison between imposing perturbation in the image space and feature space.

**Comparison with tuned weight decay.** As analysed our WFRM works by incorporating a semi-adaptive $L_2$ regularization of the classifier. Therefore we also conduct the comparison with the tuned weight decay results. By extensively tuning the weight decay of the classifier, we find the best result appears when the coefficient equals to $1.0$. From the results in Figure 3a we can see that our WFRM outperforms the best weight decay result by 1.0%, showing the effectiveness of our method. We also perform ERM without weight decay of the last layer, which results in 72.1% on average. All these results further demonstrate the effectiveness of our method.

**Weights norm v.s. empirical risk.** We also visualize the Frobenius norm of the final layer and training loss changes along the training iterations of different $L_2$ regularization variants in Figure 3b. From the results, we can see that the weight decay penalty could have a strong effect with a large coefficient such as $0.5$, or could be mitigated with small coefficient, such as $5e^{-4}$. However, we can see that our WFRM indeed regularize the weight norm as we analyzed. More interestingly, the $L_2$ regularization brought by WFRM is well aligned with the empirical risk change, demonstrating the main difference to a normal weight decay regularizer.
Computational cost comparison. SAM is very relevant to our WFRM while SFA-A is another feature perturbation DG method. Using the same device and ResNet-18 backbone, we measure the training cost of SAM, SFA-A and our WFRM, which are 2.98, 1.39, and 1.03, respectively. These values are computed based on the base ERM cost as a unit (= 1). So our WFRM costs approximately the same as ERM and can save more than 25% and 66% training costs from SAM and SFA-A.

Visualization. We use t-SNE (van der Maaten & Hinton, 2008) to visualize the features from a randomly sampled episode of TrafficSign. As shown in Figure 4, it’s clear that the test features extracted by our model are more separable than that of other methods.

Inverse visualization of the worst-case features. We use the same architecture in (Wang et al., 2022) to map features to raw images. As shown in Figure 3c, we can see that the restored images of the vanilla features are quite similar to the original images, first demonstrating the effectiveness of this visualization tool. However, we can see that the worst-case features by our WFRM have better diversity than FGSM-F in general. Especially, we can see our WFRM could lead to the scale change compared to the original input, whereas FGSM-F lacks. This result inspires us that our WFRM can actually seek very interesting feature augmentations worth exploring in future works.

5 Conclusion

In this paper, we proposed a simple yet theory-backed risk minimization method to improve DG and FSL performance. Specifically, during the model training, we conduct a minimax optimization, where in the inner loop we seek the feature perturbation to maximize the training loss and in the outer loop we minimize the training loss of the worst-case features with respect to the model parameters. We furthermore presented a theoretical analysis of our WFRM and explained why it works better than the ERM baseline. Our WFRM implicitly leads to a risk-guided $L_2$ regularization of the final classifier weights, which is inline with a recent finding (Li et al., 2022a) that model complexity plays an important role in out-of-distribution generalization. As the linear classifier and ProtoNet are widely used in DG, and FSL problems, WFRM is plug-and-play and applicable to various base DG or FSL methods. We experiment on three DG benchmarks, including PACS, VLCS, OfficeHome, and the most challenging FSL benchmark Meta-Dataset and demonstrate WFRM outperforms various DG methods and the SOTA FSL methods towards the new state-of-the-art performances in all settings.

Acknowledgement. Liming Fang is supported by the National Key R&D Program of China (Grant No.2021YFB3100700) and the National Natural Science Foundation of China (No. U22B2029, 62272228).
References


Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic training for domain generalization. 2019.


A Appendix

A.1 Proof of Propositions

Proposition 1. Suppose the loss function is a convex function of the features and the gradient of loss function with respect to feature \( \nabla_h \ell \) is \( L \)-Lipschitz continuous, where \( h = f(x) \). Then for any given feature \( h \), the maximum value of the loss function in the \( \rho \)-neighbourhood of \( h \) can be bounded by the value of loss function at \( h + \epsilon^* \triangleq h + \rho \frac{\nabla h \ell}{||\nabla h \ell||_2} \). Namely,

\[
\ell(h + \epsilon^*) \leq \max_{||\epsilon||_2 \leq \rho} \ell(h + \epsilon) \leq \ell(h) + L \rho^2 / 2.
\]

Proof of Proposition 1. Due to the convexity, the maximum value of a convex function on a compact set can only be taken at the boundary of the set. Hence, it’s clear that inner optimization problem is equivalent to the following problem.

\[
\max_{\epsilon} \quad \ell(h + \epsilon) \\
\text{s.t.} \quad ||\epsilon||_2 = \rho
\]  

(A1)

For any \( \epsilon \) that satisfies \( ||\epsilon||_2 = \rho \), by smoothness and convexity,

\[
\ell(h) + \epsilon^T \nabla_h \ell \leq \ell(h + \epsilon) \leq \ell(h) + \epsilon^* \nabla_h \ell + L \rho^2 / 2.
\]

(A2)

Denote,

\[
\epsilon^* = \arg\max_{||\epsilon||_2 = \rho} \epsilon^T \nabla_h \ell = \rho \frac{\nabla h \ell}{||\nabla h \ell||_2}.
\]

Then, take \( \epsilon^* \) to the inequality (A2) and we can get

\[
\ell(h) + \epsilon^* \nabla_h \ell \leq \ell(h + \epsilon^*)
\]

(A3)

Recall that

\[
\max_{||\epsilon||_2 = \rho} \ell(h + \epsilon) \leq \ell(h) + \epsilon^* \nabla_h \ell + L \rho^2 / 2.
\]

(A4)

Subtracting equation (A3) from equation (A4) and take advantage of the properties of the maximum, we get

\[
\ell(h + \epsilon^*) \leq \max_{||\epsilon||_2 \leq \rho} \ell(h + \epsilon)
\]  

\[
= \max_{||\epsilon||_2 \leq \rho} \ell(h + \epsilon)
\]

\[
\leq \ell(h) + L \rho^2 / 2.
\]

(A5)

\[\square\]

Proposition 2. Let us assume the labels \( y \in \{-1, 1\} \). The loss function of the logistic regression which incorporates WFRM is \( \tilde{L}_{pert} = \log[1 + e^{y(\rho \frac{||\nabla h \ell||_2}{||\nabla h \ell||_2} - y(w^T x + b))}] \).

Proof of Proposition 2. The loss function of the vanilla logistic regression is \( \tilde{L} = \log[1 + e^{y(-w^T x - b)}] \). Then, the gradient of logistic regression loss with respect to input is

\[
\nabla_x \tilde{L} = \frac{1}{1 + e^{y(-w^T x - b)}} ye^{y(-w^T x - b)} (-w).
\]

Note that the normalized gradient is \( \frac{-w}{||w||_2^2} \). Thus, the logistic regression which incorporates feature perturbation is therefore to minimize

\[
\tilde{L}_{pert} = \log[1 + e^{y||w||_2 - y(w^T x + b)}].
\]

\[\square\]
| Model  | Backbone | ILSVRC | Omni | Acract | CUB | DTD | QDraw | Fungi | Flower | Sign | COCO | Rank |
|--------|----------|--------|------|--------|-----|-----|-------|-------|--------|------|------|------|------|
| Proto  | Res34    | 53.70 ±1.07 | 68.30 ±1.27 | 58.00 ±0.06 | 74.10 ±0.92 | 68.30 ±1.77 | 53.30 ±1.06 | 40.71 ±0.15 | 87.00 ±0.73 | 58.10 ±0.05 | 41.70 ±0.08 | 6  |
| CTX    |          | 62.76 ±0.99 | 82.21 ±1.00 | 79.49 ±0.89 | 80.63 ±0.88 | 75.50 ±1.14 | 72.66 ±0.29 | 51.58 ±1.11 | 95.34 ±0.37 | 82.65 ±0.76 | 59.90 ±0.02 | 4.2 |
| TSA    |          | 63.73 ±0.99 | 82.58 ±1.11 | 80.13 ±1.01 | 83.39 ±0.90 | 79.61 ±0.68 | 71.00 ±0.34 | 51.38 ±1.17 | 94.05 ±0.45 | 81.71 ±0.95 | 61.67 ±0.95 | 4  |
| eTT    | ViT-s    | 67.37 ±0.97 | 78.11 ±1.22 | 79.94 ±1.06 | 85.93 ±0.91 | 87.62 ±0.57 | 71.34 ±0.67 | 61.80 ±1.06 | 96.37 ±0.46 | 85.09 ±0.90 | 62.33 ±0.99 | 3.2 |
| FGSM-F |          | 71.26 ±0.99 | 80.00 ±1.11 | 82.88 ±0.98 | 87.17 ±0.90 | 88.36 ±0.62 | 75.04 ±0.83 | 64.56 ±0.98 | 96.84 ±0.40 | 84.58 ±0.97 | 64.82 ±0.93 | 2.2 |
| WFRM   |          | 73.68 ±0.94 | 81.77 ±1.05 | 83.27 ±0.93 | 88.23 ±0.84 | 89.04 ±0.57 | 76.50 ±0.68 | 65.11 ±1.00 | 96.55 ±0.41 | 86.64 ±0.93 | 65.41 ±0.90 | 1.4 |

Table A1: Test accuracies, confidence interval and average rank on Meta-Dataset.

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Methods</th>
<th>Art</th>
<th>Cartoon</th>
<th>Photo</th>
<th>Sketch</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACS</td>
<td>GroupDRO</td>
<td>78.3 ±0.4</td>
<td>76.7 ±2.2</td>
<td>94.5 ±0.6</td>
<td>75.8 ±0.6</td>
<td>81.3 ±0.6</td>
</tr>
<tr>
<td>PACS</td>
<td>VREx</td>
<td>81.2 ±0.8</td>
<td>75.1 ±1.2</td>
<td>94.0 ±0.3</td>
<td>76.2 ±1.2</td>
<td>81.7 ±0.4</td>
</tr>
<tr>
<td>PACS</td>
<td>CAD</td>
<td>80.3 ±0.9</td>
<td>73.8 ±0.8</td>
<td>94.1 ±0.2</td>
<td>76.3 ±0.7</td>
<td>81.1 ±0.3</td>
</tr>
<tr>
<td>PACS</td>
<td>CausIRL</td>
<td>81.8 ±0.3</td>
<td>71.4 ±0.4</td>
<td>94.3 ±0.7</td>
<td>76.4 ±0.6</td>
<td>81.0 ±0.1</td>
</tr>
<tr>
<td>PACS</td>
<td>Transfer</td>
<td>79.4 ±0.8</td>
<td>74.5 ±1.4</td>
<td>92.5 ±0.4</td>
<td>78.4 ±0.2</td>
<td>81.2 ±0.4</td>
</tr>
<tr>
<td>PACS</td>
<td>WFRM</td>
<td>78.5 ±0.6</td>
<td>71.6 ±0.4</td>
<td>95.0 ±0.2</td>
<td>71.4 ±1.0</td>
<td>79.2 ±0.3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>V</th>
<th>L</th>
<th>C</th>
<th>S</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>70.2 ±0.9</td>
<td>61.8 ±1.0</td>
<td>97.5 ±0.7</td>
<td>73.0 ±0.6</td>
</tr>
<tr>
<td>DANN</td>
<td>70.4 ±0.7</td>
<td>62.4 ±1.1</td>
<td>96.9 ±0.6</td>
<td>74.5 ±0.4</td>
</tr>
<tr>
<td>RSC</td>
<td>69.2 ±1.4</td>
<td>63.0 ±0.9</td>
<td>96.9 ±0.2</td>
<td>72.2 ±0.9</td>
</tr>
<tr>
<td>IIMT</td>
<td>69.3 ±0.9</td>
<td>62.0 ±0.2</td>
<td>96.6 ±0.7</td>
<td>72.3 ±1.5</td>
</tr>
<tr>
<td>VLCS</td>
<td>GroupDRO</td>
<td>68.5 ±1.1</td>
<td>62.7 ±1.2</td>
<td>96.1 ±0.9</td>
</tr>
<tr>
<td>VLCS</td>
<td>VREx</td>
<td>67.0 ±0.3</td>
<td>61.8 ±1.4</td>
<td>97.0 ±0.6</td>
</tr>
<tr>
<td>VLCS</td>
<td>CAD</td>
<td>69.4 ±0.4</td>
<td>60.7 ±0.4</td>
<td>97.2 ±0.1</td>
</tr>
<tr>
<td>VLCS</td>
<td>CausIRL</td>
<td>71.0 ±0.5</td>
<td>61.6 ±1.2</td>
<td>97.0 ±0.3</td>
</tr>
<tr>
<td>VLCS</td>
<td>Transfer</td>
<td>66.8 ±0.8</td>
<td>60.3 ±0.3</td>
<td>96.1 ±0.9</td>
</tr>
<tr>
<td>VLCS</td>
<td>WFRM</td>
<td>73.6 ±0.5</td>
<td>63.4 ±0.5</td>
<td>97.9 ±0.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Art</th>
<th>Clipart</th>
<th>Product</th>
<th>Real World</th>
<th>Ave.</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERM</td>
<td>53.1 ±0.5</td>
<td>48.6 ±0.6</td>
<td>69.8 ±0.4</td>
<td>71.8 ±0.8</td>
</tr>
<tr>
<td>DANN</td>
<td>51.6 ±0.4</td>
<td>48.1 ±0.2</td>
<td>68.6 ±0.2</td>
<td>70.7 ±0.5</td>
</tr>
<tr>
<td>RSC</td>
<td>50.2 ±0.3</td>
<td>46.6 ±0.7</td>
<td>67.8 ±0.1</td>
<td>69.1 ±0.1</td>
</tr>
<tr>
<td>IIMT</td>
<td>52.4 ±2.0</td>
<td>47.8 ±0.4</td>
<td>69.1 ±0.5</td>
<td>71.0 ±0.4</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>GroupDRO</td>
<td>53.2 ±0.4</td>
<td>47.5 ±0.5</td>
<td>68.4 ±1.0</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>VREx</td>
<td>50.8 ±0.2</td>
<td>48.6 ±0.8</td>
<td>68.4 ±0.1</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>CAD</td>
<td>51.6 ±0.6</td>
<td>49.1 ±0.8</td>
<td>69.4 ±0.8</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>CausIRL</td>
<td>53.3 ±1.0</td>
<td>47.6 ±0.5</td>
<td>70.5 ±0.3</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>Transfer</td>
<td>47.0 ±0.3</td>
<td>48.3 ±0.8</td>
<td>67.1 ±0.7</td>
</tr>
<tr>
<td>OfficeHome</td>
<td>WFRM</td>
<td>55.2 ±1.2</td>
<td>49.3 ±0.0</td>
<td>70.4 ±0.4</td>
</tr>
</tbody>
</table>

Table A2: Results on DomainBed benchmark.

### A.2 Test accuracies together with confidence interval on Meta-Dataset

We show the complete results on Meta-Dataset in Table A1. In addition to the significant improvement in accuracy, the confidence interval of our WFRM is comparable to or even smaller than other methods, which indicates the model can be more robust after applying our method.
Figure A1: t-SNE visualization of learned features on PACS. Left and right show feature space of ERM and WFRM, respectively. All features are from target domain.

Figure A2: The loss curve of perturbed features derived by WFRM or PGD on VLCS. It’s clear that WFRM is able to find features with higher loss value under the same search radius.

A.3 Detailed Results on DomainBed

The detailed results on DomainBed benchmarks are presented in Table A2. It’s clear that most DG methods have difficulty in beating the vanilla REM. In contrast, our WFRM outperforms ERM by a clear margin 1.9% on VLCS and 1.0% on OfficeHome and achieves state-of-the-art on almost all the domains belonging to those two datasets. Despite the poor performance on PACS, WFRM gets the highest average accuracy. Moreover, the standard deviation obtained by WFRM tends to be smaller than the vanilla ERM, which indicates that WFRM may have the capacity of stabilizing the training process.

A.4 More Visualization

We visualize the learned feature space in domain generalization task. As shown in Figure A1, the features of different categories are not well dispersed in the feature space obtained by ERM. This phenomenon is very evident in the central part of the left figure. As a comparison, the features from different classes is much more separable by training with our WFRM.

A.5 Further Comparison with PGD

During our experiments, we also attempted to use an iterative scheme as PGD in the feature space, but it displayed poor performance. It is worth noting that PGD utilizes projected gradient descent to solve the constrained optimization problem, which may not be effective when facing a constrained concave minimization problem. To test our conjecture, we compared the loss values of perturbed features generated by our WFRM using Proposition 1 and PGD. The results on VLCS, presented in Figure A2, show that, as expected, the loss values of perturbed PGD features are smaller during the training process under the same search radius. This further confirms the significance of Proposition 1.
A.6 Quantitative Measurements of Inverse Visualization

To demonstrate the effect of WFRM more powerfully, 400 images are randomly selected from ImageNet for quantitative measurements of inverse visualization in Sec. 4.3. We utilize the reconstruction algorithm in (Wang et al., 2022) to generate the inverse images from the vanilla, FGSM perturbed, and WFRM perturbed features. And we measure the difference between their generated images and the original ones using MSE and PSNR. Tab. A3 displays the quantitative results and suggests that the features perturbed by WFRM can produce more diverse images compared with the original ones than the features perturbed by F-FGSM.

<table>
<thead>
<tr>
<th>Method</th>
<th>MSE</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reconstruct</td>
<td>3812.34</td>
<td>12.89</td>
</tr>
<tr>
<td>F-FGSM</td>
<td>4623.65</td>
<td>12.02</td>
</tr>
<tr>
<td>WFRM</td>
<td>7371.23</td>
<td>10.13</td>
</tr>
</tbody>
</table>

Table A3: Quantitative measurements for the visualization.