UNCERTAIN OUT-OF-DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We study a worst-case scenario in generalization: Out-of-domain generalization from a single source. The goal is to learn a robust model from a single source and expect it to generalize over many unknown distributions. This challenging problem has been seldom investigated while existing solutions suffer from various limitations. In this paper, we propose a new solution. The key idea is to augment the source capacity in both input and label spaces, while the augmentation is guided by uncertainty assessment. To the best of our knowledge, this is the first work to (1) access the generalization uncertainty from a single source and (2) leverage it to guide both input and label augmentation for robust generalization. The model training and deployment are effectively organized in a Bayesian meta-learning framework. We conduct extensive comparisons and ablation study to validate our approach. The results prove our superior performance in a wide scope of tasks including image classification, text classification, and speech recognition.[1]

1 INTRODUCTION

Existing machine learning algorithms have achieved remarkable success under the assumption that training and test data are sampled from similar distributions. When this assumption no longer holds, even strong models (*e.g.,* deep neural networks) may fail to produce reliable predictions. In this paper, we study a worst-case scenario in generalization: *Out-of-domain generalization from a single source.* A model learned from a single source is expected to generalize over a series of unknown distributions. This problem is more challenging than *domain adaptation* (Motiian et al., 2017a; Murez et al., 2018; Xu et al., 2019; Liu et al., 2019) which usually requires the assessment of target distributions during training, and *domain generalization* (Muandet et al., 2013; Ghifary et al., 2015; Li et al., 2018; Carlucci et al., 2019; Dou et al., 2019) which often assumes the availability of multiple sources. For example, there exists significant distribution difference in medical images collected across different hospitals. The intelligent diagnosis system is required to process images unexplored during training where model update is infeasible due to time or resource limitations.

Recently, Volpi et al. (2018) casts this problem in an ensemble framework. It learns a group of models each of which tackles an unseen test domain. This is achieved by performing *adversarial training* (Goodfellow et al., 2015) on the source to mimic the unseen test distributions. Yet, its generalization capability is limited due to the proposed semantic constraint, which allows only a small amount of data augmentation to avoid semantic changes in the label space. To address this limitation, Qiao et al. (2020) proposes *adversarial domain augmentation* to relax the constraint. By maximizing the Wasserstein distance between the source and augmentation, the domain transportation is significantly enlarged in the input space.

However, existing data (domain) augmentation based methods (Volpi et al., 2018; Qiao et al., 2020; DeVries & Taylor, 2017; Cubuk et al., 2019; Hendrycks et al., 2020) merely consider to increase the source capacity by perturbing the input space. Few of them investigate the possibility of label augmentation. An exception is Mixup (Zhang et al., 2018) which pioneers label augmentation by randomly interpolating two data examples in both input and label spaces. However, Mixup can hardly address the out-of-domain generalization problem since it is restricted in creating in-domain generations due to the linear interpolation assumption. Besides, the interpolations are randomly sampled from a fixed distribution, which also largely restricts the flexibility of domain mixtures, yielding sub-optimal performance for unseen domain generalization.

[1] We provide source codes and pre-trained models in supplementary to reproduce the reported results.

Another limitation of existing work (Muandet et al., 2013; Ghifary et al., 2015; Li et al., 2018; Carlucci et al., 2019; Dou et al., 2019) is they usually overlook the potential risk of leveraging augmented data in tackling out-of-domain generalization. This raises serious safety and security concerns in mission-critical applications (Finn et al., 2018). For instance, when deploying self-driving cars in unknown environments, it is crucial to be aware of the predictive uncertainty in risk assessment.

To tackle the aforementioned limitations, we propose uncertain out-of-domain generalization. The key idea is to increase the source capacity guided by uncertainty estimation in both input and label spaces. More specifically, in the input space, instead of directly augmenting raw data (Volpi et al., 2018; Qiao et al., 2020), we apply uncertainty-guided perturbations to latent features, yielding a domain-knowledge-free solution for various modalities such as image, text, and audio. In the label space, we leverage the uncertainty associated with feature perturbations to augment labels via interpolation, improving generalization over unseen domains. Moreover, we explicitly model the domain uncertainty as a byproduct of feature perturbation and label mixup, guaranteeing fast risk assessment without repeated sampling. Finally, we organize the training and deployment in a Bayesian meta-learning framework that is specially tailored for single source generalization. To summarize, our contribution is multi-fold:

- To the best of our knowledge, we are the first to access the uncertainty from a single source. We leverage the uncertainty assessment to gradually improve the domain generalization in a curriculum learning scheme.
- For the first time, we propose learnable label mixup in addition to widely used input augmentation, further increasing the domain capacity and reinforcing generalization over unseen domains.
- We propose a Bayesian meta-learning method to effectively organize domain augmentation and model training. Bayesian inference is crucial in maximizing the posterior of domain augmentations, such that they can approximate the distribution of unseen domains.
- Extensive comparisons and ablation study prove our superior performance in a wide scope of tasks including image classification, text classification, and speech recognition.

2 Related Work

Out-of-Domain Generalization. Domain generalization (Ghifary et al., 2015; Li et al., 2017; Grubinger et al., 2017; Shankar et al., 2018; Carlucci et al., 2019; Dou et al., 2019) has been intensively studied in recent years. JiGen (Carlucci et al., 2019) proposed to generate jigsaw puzzles from source domains and leverage them as self-supervised signals. Specially, GUD (Volpi et al., 2018) proposed adversarial data augmentation to solve single domain generalization, and learned an ensemble model for stable training. M-ADA (Qiao et al., 2020) extended it to create augmentations with large domain transportation, and designed an efficient meta-learning scheme within a single unified model. Both GUD (Volpi et al., 2018) and M-ADA (Qiao et al., 2020) fail to assess the uncertainty of augmentations and only augment the input, while our method explicitly model the uncertainty and leverage it to increase the augmentation capacity in both input and label spaces. Several methods (Madry et al., 2018; Wang et al., 2019; Hendrycks et al., 2019) proposed to leverage adversarial training (Goodfellow et al., 2015) to learn robust models, which can also be applied in single source generalization. PAR (Wang et al., 2019) proposed to learn robust global representations by penalizing the predictive power of local representations. Hendrycks et al. (2019) applied self-supervised learning to improve the model robustness.

Uncertainty Assessment. Bayesian neural networks (Hinton & Van Camp, 1993; Graves, 2011; Blundell et al., 2015) have been intensively studied to integrate uncertainty into weights of deep networks. Instead, we apply Bayesian inference to assess the uncertainty of domain augmentations. Several Bayesian meta-learning frameworks (Grant et al., 2018; Finn et al., 2018; Yoon et al., 2018; Lee et al., 2020) have been proposed to model the uncertainty of few-shot tasks. Grant et al. (2018) proposed the first Bayesian variant of *model-agnostic meta-learning* (MAML) (Finn et al., 2017) using the Laplace approximation. Finn et al. (2018) approximated MAP inference of the task-specific weights while maintain uncertainty only in the global weights. In this paper, instead of modelling the uncertainty of tasks, we propose a novel Bayesian meta-learning framework to maximize the posterior distribution of domain augmentations.

3 METHOD

We first describe our problem setting and overall framework design. The goal is to learn a robust model from a *single* domain $\mathcal{S}$ and we expect the model to generalize over an *unknown* domain distribution $\{\mathcal{T}_1, \mathcal{T}_2, \cdots\} \sim p(\mathcal{T})$. This problem is more challenging than *domain adaptation* (assuming $p(\mathcal{T})$ is given) and *domain generalization* (assuming multiple source domains $\{\mathcal{S}_1, \mathcal{S}_2, \cdots\}$ are available). We create a series of domain augmentations $\{\mathcal{S}_1^+, \mathcal{S}_2^+, \cdots\} \sim p(\mathcal{S}^+)$ to approximate $p(\mathcal{T})$, from which the backbone θ can learn to generalize over unseen domains.

Uncertainty-guided domain generalization. We assume that $\mathcal{S}^+$ should integrate uncertainty assessment for efficient domain generalization. To achieve it, we introduce the auxiliary $\psi = \{\phi_p, \phi_m\}$ to explicitly model the uncertainty with respect to θ and leverage it to create $\mathcal{S}^+$ by increasing the capacity in both input and label spaces. In input space, we introduce ϕ_p to create feature augmentations $\mathbf{h}^+$ via adding perturbation $\mathbf{e}$ sampled from $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$. In label space, we integrate the same uncertainty encoded in $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ into ϕ_m and propose learnable mixup to generate $\mathbf{y}^+$ (together with $\mathbf{h}^+$) through three variables (a, b, τ), yielding consistent augmentation in both input and output spaces. To effectively organize domain augmentation and model training, we propose a Bayesian meta-learning framework to *maximizing a posterior* of $p(\mathcal{S}^+)$ by jointly optimizing the backbone θ and the auxiliary ψ. The overall framework is shown in Fig. 1.

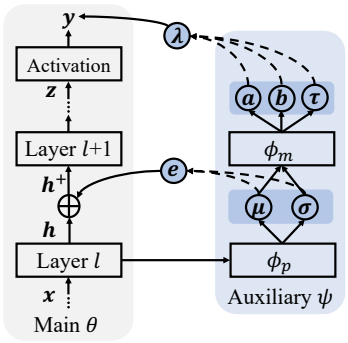

Figure 1: The main and auxiliary models.

Merits of uncertainty assessment. Assessing the uncertainty of $\mathcal{S}^+$ plays a key role in our design. First, it provides consistent guidance to the augmentation in both input and label spaces when inferencing $\mathcal{S}^+$, which has never been studied before. Second, we can gradually enlarge the domain transportation by increasing the uncertainty of $\mathcal{S}^+$ in a curriculum learning scheme (Bengio et al., 2009). Last, we can easily assess the domain uncertainty by checking the value of $\boldsymbol{\sigma}$, which measures how unsure it is when deploying on unseen domains $\mathcal{T}$ (Sec. 3.3).

3.1 UNCERTAINTY-GUIDED INPUT AUGMENTATION

The goal is to create $\mathcal{S}^+$ from $\mathcal{S}$ such that $p(\mathcal{S}^+)$ can approximate the out-of-domain distribution of $\mathcal{S}$. One the one hand, we expect a large domain transportation from $\mathcal{S}$ to $\mathcal{S}^+$ to best accommodate the unseen testing distribution $p(\mathcal{T})$. On the other hand, we prefer the transportation is domain-knowledge-free with uncertainty guarantee for broad and safe domain generalization. Towards this goal, we introduce ϕ_p to create feature augmentation $\mathbf{h}^+$ with large domain transportation through increasing the uncertainty with respect to θ.

Adversarial Domain Augmentation. To encourage large domain transportation, we cast the problem in a worst-case scenario (Sinha et al., 2018) and propose to learn the auxiliary mapping ϕ_p via *adversarial domain augmentation*:

$$\underset{\phi_p}{\text{maximize}} \underbrace{\mathcal{L}(\theta; \mathcal{S}^+)}_{\text{Main task}} - \beta \underbrace{\left\|\mathbf{z} - \mathbf{z}^+\right\|_2^2}_{\text{Constraint}}. \tag{1}$$

Here, $\mathcal{L}$ denotes empirical loss such as cross-entropy loss for classification. The second term is the worst-case constraint, bounding the largest domain discrepancy between $\mathcal{S}$ and $\mathcal{S}^+$ in embedding space. $\mathbf{z}$ denotes the FC-layer output right before the activation layer, which is distinguished from $\mathbf{h}$ that denotes the Conv-layer outputs. One merit of the proposed uncertainty-guided augmentation is that we can effectively relax the constraint to encourage large domain transportation in a curriculum learning scheme, which is significantly more efficient than Qiao et al. (2020) that has to train an extra WAE-GAN (Tolstikhin et al., 2018) to achieve this goal.

Variational feature perturbation. To achieve domain-knowledge-free augmentation, we apply uncertainty-guided perturbations to latent features instead of directly augmenting raw data. We

propose to learn layer-wise feature perturbations $\mathbf{e}$ that transport latent features $\mathbf{h} \rightarrow \mathbf{h}^+$ for efficient domain augmentation $\mathcal{S} \rightarrow \mathcal{S}^+$. Instead of a direct generation $\mathbf{e} = f_{\phi_p}(\mathbf{x}, \mathbf{h})$ widely used in previous work (Volpi et al., 2018; Qiao et al., 2020), we assume $\mathbf{e}$ follows a multivariate Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$, which can be used to easily access the uncertainty. The Gaussian parameters are learnable via variational inference $(\boldsymbol{\mu}, \boldsymbol{\sigma}) = f_{\phi_p}(\mathcal{S}, \theta)$, such that:

$$\mathbf{h}^+ \leftarrow \mathbf{h} + \text{Softplus}(\mathbf{e}), \text{ where } \mathbf{e} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}), \tag{2}$$

where $\text{Softplus}(\cdot)$ is applied to stabilize the training. ϕ_p can create a series of feature augmentations $\{\mathbf{h}_1^+, \mathbf{h}_2^+, \cdots\}$ in different training iterations. In Sec. 4.4, we empirically show that $\{\mathbf{h}_1^+, \mathbf{h}_2^+, \cdots\}$ gradually enlarge the transportation through increasing the uncertainty of augmentations in a curriculum learning scheme and enable the model to learn from "easy" to "hard" domains.

3.2 Uncertainty-Guided Label Mixup

Feature perturbations not only augment the input but also yield label uncertainty. To explicitly model the label uncertainty, we leverage the input uncertainty, encoded in $(\boldsymbol{\mu}, \boldsymbol{\sigma})$, to inference the label uncertainty encoded in (a, b, τ) through ϕ_m as shown in Fig. 1. We leverage the label uncertainty to propose learnable label mixup, yielding consistent augmentation in both input and output spaces and further reinforcing generalization over unseen domains.

Random Mixup. We start by introducing random *mixup* (Zhang et al., 2018) for robust learning. The key idea is to regularize the training to favor simple linear behavior in-between examples. More specifically, *mixup* performs training on convex interpolations of pairs of examples $(\mathbf{x}_i, \mathbf{x}_j)$ and their labels $(\mathbf{y}_i, \mathbf{y}_j)$:

$$\mathbf{x}^+ = \lambda \mathbf{x}_i + (1-\lambda)\mathbf{x}_j, \quad \mathbf{y}^+ = \lambda \mathbf{y}_i + (1-\lambda)\mathbf{y}_j, \quad \lambda \sim \text{Beta}(\alpha, \alpha), \tag{3}$$

where the *mixup* hyper-parameter $\alpha \in (0, +\infty)$ controls the interpolation strength.

Learnable Label Mixup. We cast *mixup* in a learnable framework which is specially tailored for single source generalization. First, instead of mixing up pairs of examples, we mix up $\mathcal{S}$ and $\mathcal{S}^+$ to achieve in-between domain interpolations. Second, we leverage the uncertainty encoded in $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ to predict learnable parameters (a, b), which controls the direction and strength of domain interpolations:

$$\mathbf{h}^+ = \lambda \mathbf{h} + (1-\lambda)\mathbf{h}^+, \quad \mathbf{y}^+ = \lambda \mathbf{y} + (1-\lambda)\tilde{\mathbf{y}}, \quad \lambda \sim \text{Beta}(a, b), \tag{4}$$

where $\tilde{\mathbf{y}}$ denotes a *label-smoothing* (Szegedy et al., 2016) version of $\mathbf{y}$. More specifically, we perform *label smoothing* by a chance of τ, such that we assign $\rho \in (0, 1)$ to the true category and equally distribute $\frac{1-\rho}{c-1}$ to the others, where c counts categories. The Beta distribution (a, b) and the lottery τ are jointly inferred by $(a, b, \tau) = f_{\phi_m}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ to integrate the uncertainty of domain augmentation.

3.3 Bayesian Meta-learning

We propose to model all variables $\{\boldsymbol{\mu}, \boldsymbol{\sigma}, a, b, \tau\}$ via Bayesian inference sharing the same auxiliary model $\psi = \{\phi_p, \phi_m\}$ for efficient domain augmentation and model training. Bayesian inference is crucial in *maximizing a posterior* of $p(\mathcal{S}^+)$ such that it can approximates $p(\mathcal{T})$ which is unknown during training. More importantly, we can solve such MAP problem efficiently by utilizing a modified *model-agnostic meta-learning* (MAML) (Finn et al., 2017) framework to distill and transfer meta-knowledge sourcing from the prior $\mathcal{S}$. Then, we can meta-learn the initial backbone θ to generalize over the approximated populations $p(\mathcal{S}^+)$, such that generalization errors are bounded in a worst-case scenario when deploying $\{\theta, \psi\}$ on unseen populations $p(\mathcal{T})$. The gradient computation is:

$$\mathbf{g}^+ = \nabla_{\theta,\psi} \mathbb{E}_{p(\mathcal{S}^+)}[\mathcal{L}(\theta^*; \mathcal{S}^+)], \text{ where } \theta^* \equiv \theta - \alpha \nabla_\theta \mathcal{L}(\theta; \mathcal{S}). \tag{5}$$

Here, α denotes the learning rate.

Bayesian Meta-learning. The goal is to maximize the conditional likelihood of the augmented domain $\mathcal{S}^+$: $\log p\left(\mathbf{y}^+ | \mathbf{x}, \mathbf{h}^+; \theta^*\right)$. However, solving it involves the true posterior $p\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$, which is intractable (Lee et al., 2020). Thus, we resort to amortized variational inference with a tractable form of approximate posterior $q\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$. The approximated lower bound is as follows:

$$L_{\theta,\psi} = \mathbb{E}_{q(\mathbf{h}^+|\mathbf{x};\theta^*,\psi)}[\log \frac{p\left(\mathbf{y}^+ | \mathbf{x}, \mathbf{h}^+; \theta^*\right)}{q\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)}] \leq \log p\left(\mathbf{y}^+ | \mathbf{x}, \mathbf{h}^+; \theta^*\right) \tag{6}$$

We leverage Monte-Carlo (MC) sampling to maximize the lower bound $L_{\theta,\psi}$ by:

$$\min_{\theta,\psi} \frac{1}{K} \sum_{k=1}^{K} \left[-\log p\left(\mathbf{y}_k^+ | \mathbf{x}, \mathbf{h}_k^+; \theta^*\right) \right] + \mathrm{KL}\left[q\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right) \| p\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right) \right], \tag{7}$$

where $\mathbf{h}_k^+ \sim q\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$ and K is the number of MC samples. Instead of setting $p\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$ to $\mathcal{N}(\mathbf{0}, \mathbf{I})$ in (Kingma & Welling, 2014), we assume that $q\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$ can approximate $p\left(\mathbf{h}^+ | \mathbf{x}; \theta^*, \psi\right)$ through the adversarial training on ϕ_p in Eq. 1.

Uncertainty Estimation. At testing time, given a novel domain $\mathcal{T}$, we propose a *normalized domain uncertainty score*, $|\frac{\boldsymbol{\sigma}(\mathcal{T}) - \boldsymbol{\sigma}(\mathcal{S})}{\boldsymbol{\sigma}(\mathcal{S})}|$, to estimate its uncertainty with respect to learned θ. Considering ψ is usually much smaller than θ, this score can be calculated efficiently by one-pass data forwarding through ψ. In Fig. 2, we empirically prove that our estimation is consistent with conventional Bayesian methods (Blundell et al., 2015), while the time consumption is significantly reduced by an order of magnitude.

4 Experiments

To best validate the performance, we conduct a series of experiments to compare our approach with existing methods that can be roughly grouped in four categories: **1) Adversarial training:** PAR (Wang et al., 2019), Self-super (Hendrycks et al., 2019), and PGD (Madry et al., 2018). **2) Data augmentation:** Mixup (Zhang et al., 2018), JiGen (Carlucci et al., 2019), Cutout (DeVries & Taylor, 2017), and AutoAug (Cubuk et al., 2019). **3) Domain adaptation:** DIRT-T (Shu et al., 2018), SE (French et al., 2018), SBADA (Russo et al., 2018), FADA (Motiian et al., 2017a), and CCSA (Motiian et al., 2017b). **4) Domain generalization:** ERM (Koltchinskii, 2011), GUD (Volpi et al., 2018), and M-ADA (Qiao et al., 2020). The experimental results prove that our method achieves superior performance on a wide scope of tasks, including *image classification* (Hendrycks & Dietterich, 2019), *text classification* (Chen et al., 2012), and *speech recognition* (Warden, 2018). Please refer to Appendix B for more details about experiment setup.

4.1 Image Classification

Table 1: Image classification accuracy (%) on *Digits* (Volpi et al., 2018) (**top**) and *CIFAR-10-C* (Hendrycks & Dietterich, 2019) (**bottom**). We compare with *robust training* (**Columns 1-4**) and *domain generalization* (**Columns 5-7**). For *Digits*, all models are trained on *MNIST* (LeCun et al., 1998). For *CIFAR-10-C*, two widely employed backbones are evaluated. Our method outperforms M-ADA (Qiao et al., 2020) (previous SOTA) consistently in all settings.

Domain	Mixup	PAR	Self-super	JiGen	ERM	GUD	M-ADA	**Ours**
SVHN	28.5	30.5	30.0	33.8	27.8	35.8	42.1	**43.3**
MNIST-M	54.0	58.4	58.1	57.8	52.8	60.7	66.8	**67.4**
SYN	41.2	44.1	41.9	43.8	39.9	45.0	50.4	**57.1**
USPS	76.6	76.9	77.1	77.2	76.5	77.3	77.1	**77.4**
Avg.	50.1	52.5	51.8	53.1	49.3	54.7	59.1	**61.3**

Model	Mixup	Cutout	AutoAug	PGD	ERM	GUD	M-ADA	**Ours**
AllConv	75.4	67.1	70.8	71.9	69.2	73.6	75.9	**79.6**
WRN	77.7	73.2	76.1	73.8	73.1	75.3	80.2	**83.4**

Datasets. We validate our method on the following two benchmark datasets for image classification. (1) *Digits* is used for digit classification and consists of five sub-datasets: MNIST (LeCun et al., 1998), MNIST-M (Ganin & Lempitsky, 2015), SVHN (Netzer et al., 2011), SYN (Ganin & Lempitsky, 2015), and USPS (Denker et al., 1989). Each sub-dataset can be viewed as a different domain. Each image in these datasets contains one single digit with different styles and backgrounds. (2) *CIFAR-10-C* (Hendrycks & Dietterich, 2019) is a robustness benchmark consisting of 19 corruptions types with five levels of severity applied to the test set of CIFAR-10 (Krizhevsky et al., 2009).

Setup. *Digits*: following the experimental setup in Volpi et al. (2018), we use 10,000 samples in the training set of MNIST for training, and evaluate models on the other four sub-datasets. We use a

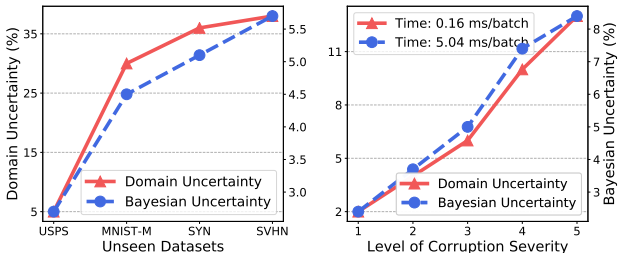

Figure 2: Uncertainty estimation on *Digits* (**left**) and *CIFAR-10-C* (**right**). Our prediction of *domain uncertainty* is consistent with *Bayesian uncertainty*, while our method is an order of magnitude faster since we forward data only once.

Method	$\lvert\mathcal{T}\rvert$	M $\rightarrow$ S	Avg.
DIRT-T	All	54.5	-
SE		14.0	70.4
SBADA		**61.1**	78.3
FADA	7	47.0	75.2
CCSA	10	37.6	76.0
Ours	7	58.1	80.1
	10	59.8	**81.5**

Table 2: Few-shot domain adaptation accuracy (%) on *MNIST(M), USPS(U), and SVHN(S)*. $\lvert\mathcal{T}\rvert$ denotes the number of target samples (per class) used during model training.

ConvNet (LeCun et al., 1989) with architecture *conv-pool-conv-pool-fc-fc-softmax* as the backbone. All images are resized to 32×32, and the channels of MNIST and USPS are duplicated to make them as RGB images. *CIFAR-10-C*: we train models on CIFAR-10 and evaluate them on CIFAR-10-C. Following the setting of Hendrycks et al. (2020), we evaluate the model on 15 corruptions. We train models on AllConvNet (AllConv) (Salimans & Kingma, 2016) and Wide Residual Network (WRN) (Zagoruyko & Komodakis, 2016) with 40 layers and the width of 2.

Results. 1) Classification accuracy. Tab. 1 shows the classification results of *Digits* and *CIFAR-10-C*. On the experiment of *Digits*, GUD (Volpi et al., 2018), M-ADA (Qiao et al., 2020), and our method outperform all baselines of the second block. And our method outperforms M-ADA (Qiao et al., 2020) on *SYN* and the average accuracy by 6.7% and 2.2%, respectively. On the experiment of *CIFAR-10-C*, our method consistently outperforms all baselines on two different backbones, suggesting its strong generalization on various image corruptions. **2) Uncertainty estimation.** We compare the proposed *domain uncertainty score* (Sec.3.3) with a more time-consuming one based on Bayesian models (Blundell et al., 2015). The former computes the uncertainty through one-pass forwarding, while the latter computes the variance of the output through repeated sampling of 30 times. Fig. 2 show the results of uncertainty estimation on *Digits* and *CIFAR-10-C*. As seen, our estimation shows consistent results with Bayesian uncertainty estimation on both *Digits* and *CIFAR-10-C*, suggesting its high efficiency. **3) Few-shot domain adaptation.** Although our method is designed for single domain generalization, we also show that our method can be easily applied for few-shot domain adaptation (Motiian et al., 2017a) due to the meta-learning training scheme. Following the setup in Qiao et al. (2020), the model is first pre-trained on the source domain $\mathcal{S}$ and then fine-tuned on the target domain $\mathcal{T}$. We conduct three few-shot domain adaption tasks: *USPS(U)→MNIST(M)*, *MNIST(M)→SVHN(S)*, and *SVHN(S)→MNIST(M)*. Results of the three tasks are shown in Tab. 2. Our method achieves the best performance on the average of three tasks. The result on the hardest task (*M→S*) is even competitive to that of SBADA (Russo et al., 2018) which takes advantage of all images of the target domain for training. Full results are provided in Tab. 8 (Appendix C).

4.2 Text Classification

Datasets. *Amazon Reviews* (Chen et al., 2012) contains reviews of products belonging to four categories - books(b), DVD(d), electronics(e) and kitchen appliances(k). The difference in textual description of the four product categories manifests as domain shift. Following Ganin & Lempitsky (2015), we use unigrams and bigrams as features resulting in 5000 dimensional vector representations.

Setup. We train the models on one source domain (books or dvd), and evaluate them on the other three domains. Similar to Ganin & Lempitsky (2015), we use a neural network with two hidden layers (both with 50 neurons) as the backbone.

Results. Tab. 9 shows the results of text classification on *Amazon Reviews* (Chen et al., 2012) . It appears that our method outperform previous ones on all the three unseen domains when the source domain is "books". We note that there is a little drop in performance on "electronics" when the source

domain is "dvd". One possible reason is that "electronics" and "dvd" may share a similar distribution. And our method creates large distribution shift, degrading the performance on "electronics".

4.3 SPEECH RECOGNITION

Datasets. *Google Commands* (Warden, 2018) contains 65000 utterances (one second long) from thousands of people. The goal is to classify them to 30 command words. There are 56196, 7477, and 6835 examples for training, validation, and test. To simulate domain shift in real-world scenario, we apply five common corruptions in both time and frequency domains. This creates five test sets that are "harder" than training sets, namely amplitude change (Amp.), pitch change (Pit.), background noise (Noise), stretch (Stretch), and time shift (Shift).

Setup. We train the models on the clean train set, and evaluate them on the corrupted test sets. We encode each audio into a mel-spectrogram with the size of 1x32x32 and feed them to LeNet (Lecun et al., 1998) as one-channel input.

Results. Tab. 4 shows the results of speech recognition on *Google Commands* (Warden, 2018). Our method outperforms the other three methods on all the five corrupted test sets, indicating its strong generalization ability in both time and frequency domain.

Table 3: Text classification accuracy (%) on *Amazon Reviews*. Models are trained on one text domain and evaluated on unseen text domains. Our method outperforms others in all settings except "$dvd \rightarrow electronics$".

	books			dvd		
Method	d	k	e	b	k	e
ERM	78.7	74.6	63.6	78.5	82.1	**75.2**
GUD	79.1	75.6	64.7	78.1	82.0	74.6
M-ADA	79.4	76.1	65.3	78.8	82.6	74.3
Ours	**80.2**	**76.8**	**67.1**	**80.1**	**83.5**	75.0

Table 4: Speech recognition accuracy (%) on *Google Commands*. Models are trained on clean set and evaluated on five corrupted sets. Results validate our strong generalization on corruptions in both time and frequency domains.

	Time		Frequency		
Method	Amp.	Pit.	Noise	Stretch	Shift
ERM	63.8	71.6	73.9	72.9	70.5
GUD	64.1	72.1	74.8	73.1	70.9
M-ADA	64.5	71.9	75.4	73.8	71.4
Ours	**65.3**	**73.5**	**75.8**	**75.0**	**72.5**

4.4 ABLATION STUDY

In this section, we perform ablation study to investigate key components of our method. For *Digits* (Volpi et al., 2018), we report the average performance of all unseen domains. For *CIFAR-10-C* (Hendrycks & Dietterich, 2019), we report the average performance of all types of corruptions at the highest level of severity.

Uncertainty assessment. We visualize feature perturbation $|\mathbf{e}| = |\mathbf{h}^+ - \mathbf{h}|$ and the embedding of domains at different training iterations T on MNIST (LeCun et al., 1998). We use t-SNE (Maaten & Hinton, 2008) to visualize the source and augmented domains without and with uncertainty assessment in the embedding space. Results are shown in Fig. 3. In the model without uncertainty (left), the feature perturbation $\mathbf{e}$ is sampled from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ without learnable parameters. In the model with uncertainty (right), we observe that most perturbations are located in the background area which increases the variation of $\mathcal{S}^+$ while keeping the category unchanged. As a result, models with uncertainty can create large domain transportation in a curriculum learning scheme, yielding safe augmentation and improved accuracy on unseen domains. We visualize the density of $\mathbf{y}^+$ in Fig. 4. As seen, models with uncertainty can significantly augment the label space.

Variational feature perturbation. We investigate different designs of feature perturbation: *1) Random Gaussian:* the feature perturbation $\mathbf{e}$ is sampled from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ without learnable parameters. *2) Deterministic perturbation:* we directly add the learned $\boldsymbol{\mu}$ to $\mathbf{h}$ without sampling, yielding $\mathbf{h}^+ \leftarrow \mathbf{h} + \text{Softplus}(\boldsymbol{\mu})$. *3) Random $\boldsymbol{\mu}$:* the feature perturbation $\mathbf{e}$ is sampled from $\mathcal{N}(\mathbf{0}, \boldsymbol{\sigma})$, where $\boldsymbol{\mu} = \mathbf{0}$. *4) Random $\boldsymbol{\sigma}$:* $\mathbf{e}$ is sampled from $\mathcal{N}(\boldsymbol{\mu}, \mathbf{I})$, where $\boldsymbol{\sigma} = \mathbf{I}$. Results on these different choices are shown

Table 5: Ablation study of feature perturbation.

	Digits	CIFAR-10-C
Full Model	**61.3**±0.73	**70.2**±0.62
Random Gaussian	51.0±0.36	64.0±0.18
Determ. perturb.	59.7±0.70	67.0±0.57
Random $\boldsymbol{\mu}$	60.5±0.75	69.1±0.61
Random $\boldsymbol{\sigma}$	60.7±0.65	69.5±0.60

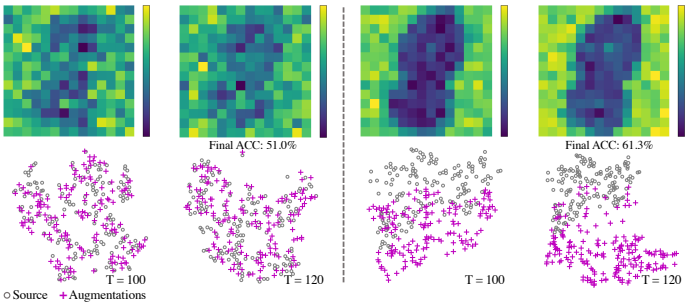

Figure 3: Visualization of feature perturbation $|\mathbf{e}| = |\mathbf{h}^+ - \mathbf{h}|$ (**Top**) and embedding of domains (**Bottom**) at different training iterations T on *MNIST*. **Left:** Models w/o uncertainty; **Right:** Models w/ uncertainty. Most perturbations are located in the background area and models w/ uncertainty can create large domain transportation in a curriculum learning scheme.

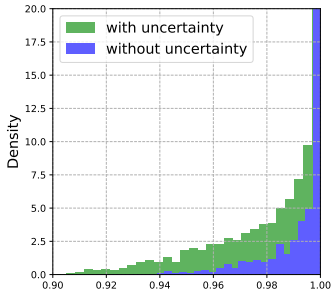

Figure 4: Density of $\mathbf{y}^+$ on *MNIST*. Models w/ uncertainty can encourage more smoothing labels and significantly increase the capacity of label space.

in Tab. 5. As seen, *Random Gaussian* yields the lowest accuracy on both datasets, indicating the necessity of learnable perturbations. *Deterministic perturbation* is inferior to *Random* $\boldsymbol{\mu}$ and *Random* $\boldsymbol{\sigma}$, suggesting that sampling-based perturbation can effectively increase the domain capacity. Finally, either *Random* $\boldsymbol{\mu}$ or *Random* $\boldsymbol{\sigma}$ is slightly worse than the full model. We conclude that both learnable $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ contribute to the final performance.

Learnable label mixup. We implement two variants of label *mixup*: *1) Without mixup:* the model is trained without label augmentation. *2) Random mixup:* the mixup coefficient λ is sampled from a fixed distribution $\mathrm{Beta}(1, 1)$. Results on the two variants are reported in Tab. 6. We notice that *Random mixup* achieves better performance than *without mixup*. The results support our claim that label augmentation can further improve the model performance. The learnable mixup (full model) achieves the best results, suggesting that the proposed learning label mixup can create informative domain interpolations for robust learning.

Table 6: Ablation study of label mixup.

	Digits	CIFAR-10-C
Full Model	**61.3**±0.73	**70.2**±0.62
w/o mixup	60.6±0.76	67.4±0.64
Random mixup	60.9±1.10	69.4±0.58

Training strategy. At last, we compare different training strategies. *1) Without adversarial training:* models are learned without adversarial training (Eq. 1). *2) Without minimizing ϕ_p:* ϕ_p is not optimized in Eq. 7. Results are reported in Tab. 7. The adversarial training contributes most to the improvements: 9.5% on *Digits* and 10.2% on *CIFAR-10-C*. We notice that the accuracy is slightly dropped without minimization of ϕ_p, possibly due to the excessive accumulation of perturbations.

Table 7: Ablation study of training strategy.

	Digits	CIFAR-10-C
Full Model	**61.3**±0.73	**70.2**±0.62
w/o adv. training	51.8±0.71	60.0±0.55
w/o minimizing ϕ_p	60.6±0.91	69.6±0.75

5 Conclusion

In this work, we introduced uncertain out-of-domain generalization to tackle the problem of single source generalization. Our method explicitly model the uncertainty of domain augmentations in both input and label spaces. In input space, the proposed uncertainty-guided feature perturbation resolves the limitation of raw data augmentation, yielding a domain-knowledge-free solution for various modalities. In label space, the proposed uncertainty-guided label mixup further increases the domain capacity. Finally, the proposed Bayesian meta-learning framework can maximize the posterior distribution of domain augmentations, such that the learned model can generalize well on unseen domains. The experimental results prove that our method achieves superior performance on a wide scope of tasks, including *image classification*, *text classification*, and *speech recognition*.

REFERENCES

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, pp. 41–48, 2009.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *ICML*, pp. 1613–1622, 2015.

Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, pp. 2229–2238, 2019.

Minmin Chen, Zhixiang Xu, Kilian Q Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *ICML*, pp. 1627–1634, 2012.

Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *CVPR*, pp. 113–123, 2019.

John S Denker, WR Gardner, Hans Peter Graf, Donnie Henderson, Richard E Howard, W Hubbard, Lawrence D Jackel, Henry S Baird, and Isabelle Guyon. Neural network recognizer for hand-written zip code digits. In *NeurIPS*, pp. 323–331, 1989.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*, pp. 6447–6458, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, pp. 1126–1135, 2017.

Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. In *NeurIPS*, pp. 9516–9527, 2018.

Geoffrey French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *ICLR*, 2018.

Yaroslav Ganin and Victor Lempitsky. Unsupervised Domain Adaptation by Backpropagation. In *ICML*, pp. 1180–1189, 2015.

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, pp. 2551–2559, 2015.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. In *ICLR*, 2018.

Alex Graves. Practical variational inference for neural networks. In *NeurIPS*, pp. 2348–2356, 2011.

Thomas Grubinger, Adriana Birlutiu, Holger Schöner, Thomas Natschläger, and Tom Heskes. Multi-domain transfer component analysis for domain generalization. *Neural Processing Letters (NPL)*, 46(3):845–855, 2017.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *ICLR*, 2019.

Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. In *NeurIPS*, pp. 15637–15648, 2019.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *COLT*, pp. 5–13, 1993.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pp. 448–456, 2015.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *arXiv:1412.6980 [cs.LG]*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

Vladimir Koltchinskii. *Oracle Inequalities in Empirical Risk Minimization and Sparse Recovery Problems: Ecole d'Eté de Probabilités de Saint-Flour XXXVIII-2008*, volume 2033. Springer Science & Business Media, 2011.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation (NC)*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Yann Lecun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11), 1998.

Hae Beom Lee, Hayeon Lee, Donghyun Na, Saehoon Kim, Minseop Park, Eunho Yang, and Sung Ju Hwang. Learning to balance: Bayesian meta-learning for imbalanced and out-of-distribution tasks. In *ICLR*, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, Broader and Artier Domain Generalization. In *ICCV*, pp. 5542–5550, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to Generalize: Meta-Learning for Domain Generalization. In *AAAI*, 2018.

Hong Liu, Mingsheng Long, Jianmin Wang, and Michael Jordan. Transferable adversarial training: A general approach to adapting deep classifiers. In *ICML*, pp. 4013–4022, 2019.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research (JMLR)*, 9(Nov):2579–2605, 2008.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. In *NeurIPS*, pp. 6670–6680, 2017a.

Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified Deep Supervised Domain Adaptation and Generalization. In *ICCV*, pp. 5715–5725, 2017b.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain Generalization via Invariant Feature Representation. In *ICML*, pp. 10–18, 2013.

Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *CVPR*, pp. 4500–4509, 2018.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Fengchun Qiao, Long Zhao, and Xi Peng. Learning to learn single domain generalization. In *CVPR*, 2020.

Paolo Russo, Fabio M Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *CVPR*, pp. 8099–8108, 2018.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NeurIPS*, pp. 901–909, 2016.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing Across Domains via Cross-Gradient Training. In *ICLR*, 2018.

Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. In *ICLR*, 2018.

Aman Sinha, Hongseok Namkoong, and John Duchi. Certifying distributional robustness with principled adversarial training. In *ICLR*, 2018.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pp. 2818–2826, 2016.

I Tolstikhin, O Bousquet, S Gelly, and B Schölkopf. Wasserstein auto-encoders. In *ICLR*, 2018.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, pp. 5334–5344, 2018.

Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, pp. 10506–10518, 2019.

Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Xiang Xu, Xiong Zhou, Ragav Venkatesan, Gurumurthy Swaminathan, and Orchid Majumder. d-sne: Domain adaptation using stochastic neighborhood embedding. In *CVPR*, pp. 2497–2506, 2019.

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. In *NeurIPS*, pp. 7332–7342, 2018.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

APPENDIX

A SOURCE CODE

We provide the source code in the folder of "**code**". To reproduce our results on *Digits* dataset (Volpi et al., 2018), please run:

sh main.sh

Details of other files and folders are as follow:

- **data:** Training set *MNIST* (LeCun et al., 1998) and test sets of *SVHN* (Netzer et al., 2011), *MNIST-M* (Ganin & Lempitsky, 2015), *SYN* (Ganin & Lempitsky, 2015), and *USPS* (Denker et al., 1989).
- **models:** definition of the backbone and auxiliary models and pre-trained models.
- **utils:** loss functions and other supportive functions.
- **train.py:** the source code to train our model.
- **test.py:** the source code to deploy the trained model.

The code is built on Pytorch 1.1.0 with CUDA 10.0. MetaNN 0.1.5 is used to implement meta-learning and Scipy 1.2.1 is used to process images. We use one GeForce RTX 2080 Ti GPU with 12G memory on Ubuntu 18.04.

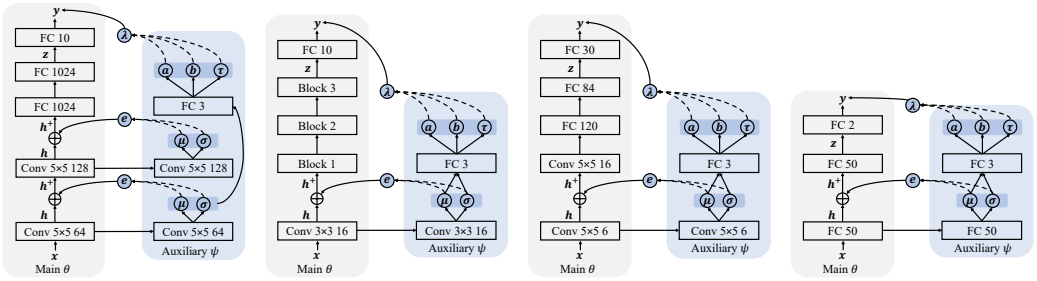

Figure 5: Architectures of main and auxiliary models. **From left to right:** (a) *Digits* (Volpi et al., 2018); (b) *CIFAR-10-C* (Hendrycks & Dietterich, 2019); (c) *Google Commands* (Warden, 2018), and (d) *Amazon Reviews* (Chen et al., 2012).

B Architecture Design and Setup

We provide more experimental details on the four datasets: *Digits* (Volpi et al., 2018), *CIFAR-10-C* (Hendrycks & Dietterich, 2019), *Amazon Reviews* (Chen et al., 2012), and *Google Commands* (Warden, 2018). In learnable label mixup (Sec. 3.2), we use Gaussian parameters of feature perturbations from the first layer. We choose specific backbone models, and design different auxiliary models as well as training strategies according to characteristics of each dataset.

In *Digits* (Volpi et al., 2018), the backbone model is *conv-pool-conv-pool-fc-fc-softmax*. There are two 5×5 convolutional layers with 64 and 128 channels respectively. Each convolutional layer is followed by a max pooling layer with the size of 2×2. The size of the two Fully-Connected (FC) layers is 1024 and the size of the softmax layer is 10. We inject perturbations to latent features of the two convolutional layers. The detailed architecture is presented in Fig. 5 (a). We employ Adam (Kingma & Ba, 2014) for optimization with batch size of 32. We train for total 10K iterations with learning rate of 10^{-4}.

In *CIFAR-10-C* (Hendrycks & Dietterich, 2019), we evaluate our method on two backbones: AllConvNet (AllConv) (Salimans & Kingma, 2016) and Wide Residual Network (WRN) (Zagoruyko & Komodakis, 2016) with 40 layers and the width of 2. In AllConv (Salimans & Kingma, 2016), the model starts with three 3×3 convolutional layers with 96 channels. Each layer is followed by batch normalization (BN) (Ioffe & Szegedy, 2015) and GELU. They convert the original image with three channels to feature maps of 96 channels. Then, the features go though three 3×3 convolutional layers with 192 channels. After that, the features are fed into two 1×1 convolutional layers with 192 channels and an average pooling layer with the size of 8×8. Finally, a softmax layer with the size of 10 is used for classification. In WRN (Zagoruyko & Komodakis, 2016). The first layer is a 3×3 convolutional layer. It converts the original image with three channels to feature maps of 16 channels. Then the features go through three blocks of 3×3 convolutional layers. Each block consists of six basic blocks and each basic block is composed of two convolutional layers with the same number of channels. And their channels are $\{32, 64, 128\}$ respectively. Each layer is followed by batch normalization (BN) (Ioffe & Szegedy, 2015). An average pooling layer with the size of 8×8 is appended to the output of the third block. Finally, a softmax layer with the size of 10 is used for prediction. In both AllConv (Salimans & Kingma, 2016) and WRN (Zagoruyko & Komodakis, 2016), we only inject perturbations to the latent features of the first convolutional layer. We also tried to inject perturbations in the next few layers or blocks, however, we found the performance degraded severely mainly due to its large effect on the semantic feature, *i.e.*, outputs before the activation layer. The detailed architecture with backbone of WRN is shown in Fig. 5 (b). Following the training procedure in Zagoruyko & Komodakis (2016), we use SGD with Nesterov momentum and set the batch size to 128. The initial learning rate is 0.1 with a linear decay and the number of epochs is 200.

In *Amazon Reviews* (Chen et al., 2012), reviews are assigned binary labels - 0 if the rating of the product is up to 3 stars, and 1 if the rating is 4 or 5 stars. The extracted features are fed into two FC layers with the size of 50. A softmax layer with the size of two is used to classify the sentiment of reviews into "positive" or "negative". All models are trained using Adam (Kingma & Ba, 2014) optimizer with learning rate of 10^{-4} and batch size of 32 for 1000 iterations. In *Google Commands* (Warden, 2018), the mel-spectrogram features are fed into LeNet (Lecun et al., 1998) as one-channel input. The original image is fed into two 5×5 convolutional layers with the channels

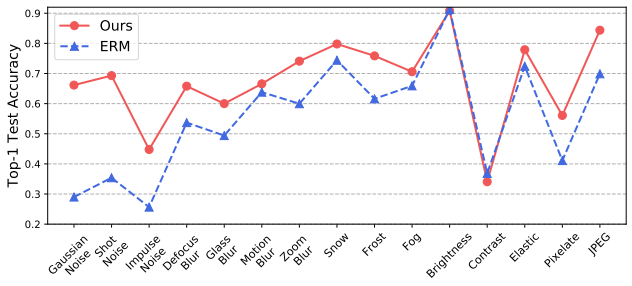

Figure 6: Classification accuracy on fifteen corruptions of *CIFAR-10-C* using the backbone of WRN (40-2). Following Fig. 7, the accuracy of each corruption with the highest level of severity is presented. Our method achieves 20% improvements on corruptions of noise.

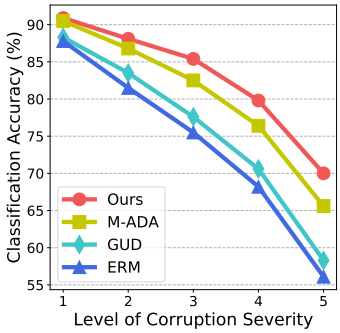

Figure 7: Classification accuracy (%) on five levels of corruption severity. Our method has the smallest degradation under the highest level of corruption severity.

of 6 and 16, respectively. Next, the features go through two FC layers with the size of 120 and 84, respectively. Finally, a softmax layer with the size of 30 is leveraged to predict the spoken word. Models are trained using Adam (Kingma & Ba, 2014) with learning rate 10^{-4} and batch size of 128 for 30 epoches. For the corrupted test sets, the range of "amplitude change" is (0.7,1.1). The maximum scales of "pitch change", "background noise", and "stretch" are 0.2, 0.45, and 0.2, respectively. The maximum shift of "time shift" is 8. In the experiments of *Amazon Reviews* (Chen et al., 2012) and *Google Commands* (Warden, 2018), feature perturbations are appended to the first layer. The detailed architectures used for *Google Commands* (Warden, 2018) and *Amazon Reviews* (Chen et al., 2012) are presented in Fig. 5 (c) and (d), respectively.

C Additional Results

C.1 Image Classification

1) Classification accuracy on *CIFAR-10-C* (Hendrycks & Dietterich, 2019). We train all models on clean data, *i.e.*, CIFAR-10, and test them on corruption data, *i.e., CIFAR-10-C*. In this case, there are totally 15 unseen testing domains. We compare our method with the other three methods for domain generalization: ERM (Koltchinskii, 2011), GUD (Volpi et al., 2018), and M-ADA (Qiao et al., 2020). The classification results on corruptions across five levels of severity are shown in Fig. 7. As seen, our method outperforms other methods across all levels of corruption severity. Specifically, the gap between M-ADA (Qiao et al., 2020) (previous SOTA) and our method gets larger with the level of severity increasing. Fig. 6 shows more detailed comparisons of all corruptions at the highest level of severity. As seen, our method achieves substantial gains across a wide variety of corruptions, with a small drop of performance in only two corruption types: brightness and contrast. Especially, accuracies are significantly improved by 20% on corruptions of noise. The results demonstrate its strong generalization capability on severe corruptions.

2) Few-shot domain adaptation. We conduct three few-shot domain adaption tasks: *USPS(U)→MNIST(M)*, *MNIST(M)→SVHN(S)*, and *SVHN(S)→MNIST(M)*. Results of the three tasks are shown in Tab. 8. As seen, the result on the hardest task (*M→S*) is even competitive to that of SBADA (Russo et al., 2018) which requires all images of the target domain during training. Specifically, our method achieves the best performance on the average of the three tasks. Note that, when the target domain changes, our method only needs to fine-tune the pre-trained model with a few samples within a small number of iterations, while other methods have to train entirely new models.

Table 8: Few-shot domain adaptation accuracy (%) on *MNIST(M), USPS(U), and SVHN(S)*. $|\mathcal{T}|$ denotes the number of target samples (per class) used during model training.

Method	$\|\mathcal{T}\|$	U → M	M → S	S → M	Avg.
DIRT-T		-	54.50	**99.40**	-
SE	All	**98.07**	13.96	99.18	70.40
SBADA		97.60	**61.08**	76.14	78.27
FADA	7	91.50	47.00	87.20	75.23
CCSA	10	95.71	37.63	94.57	75.97
Ours	7	92.97	58.12	89.30	80.13
	10	93.16	59.77	91.67	**81.53**

Table 9: Text classification accuracy (%) on *Amazon Reviews* (Chen et al., 2012). Models are trained on one text domain and evaluated on unseen text domains. Our method outperforms others in all settings except "$dvd \rightarrow electronics$" and "$electronics \rightarrow dvd$". The possible reason is that "dvd" and "$electronics$" may share a similar distribution while our method creates large distribution shift.

	books			dvd			kitchen			electronics		
Method	d	k	e	b	k	e	b	d	e	b	d	k
ERM	78.7	74.6	63.6	78.5	82.1	**75.2**	75.4	76.0	81.2	69.4	**74.8**	83.9
GUD	79.1	75.6	64.7	78.1	82.0	74.6	74.9	76.7	81.6	68.9	74.2	84.4
M-ADA	79.4	76.1	65.3	78.8	82.6	74.3	75.2	77.3	82.3	69.0	73.7	84.8
Ours	**80.2**	**76.8**	**67.1**	**80.1**	**83.5**	75.0	**76.1**	**78.2**	**83.5**	**70.2**	74.5	**85.7**

C.2 TEXT CLASSIFICATION

We train the models on one source domain, and evaluate them on the other three domains. Tab. 9 shows the results of text classification on *Amazon Reviews* (Chen et al., 2012) . We found that our method outperform previous ones on all the three unseen domains when the source domain is "books" or "kitchen". Specially, our method outperforms ERM (Koltchinskii, 2011) by 3.5% on "books $\rightarrow$ electronics". We observe that there is a little drop in accuracy on "dvd $\rightarrow$ electronics" and "electronics $\rightarrow$ dvd". One possible reason is that "electronics" and "dvd" may share a similar distribution. And our method creates large distribution shift, degrading the performance on them.

C.3 ABLATION STUDY

We study the effect of two important hyper-parameters of our model: the number of Monte-Carlo (MC) samples (K) and the coefficient of constraint (β). We report the average accuracy on the four unseen domains (*MNIST-M* Ganin & Lempitsky (2015), *SVHN* Netzer et al. (2011), *SYN* Ganin & Lempitsky (2015), and *USPS* Denker et al. (1989)). We present the classification results under different hyper-parameters in Fig. 8.

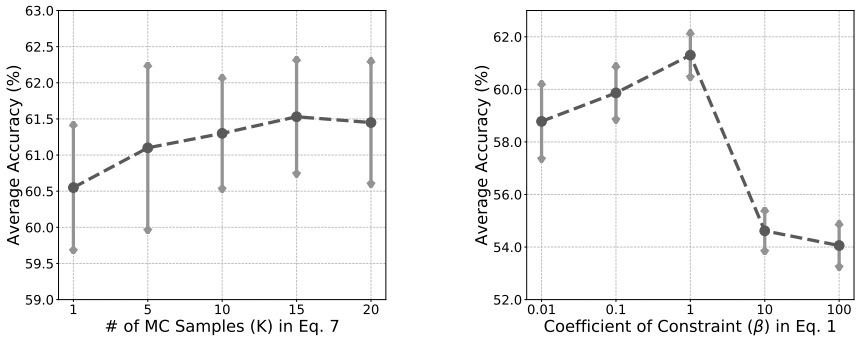

Figure 8: Ablation study on hyper-parameters K and β. The average accuracy on the four unseen domains (*MNIST-M* (Ganin & Lempitsky, 2015), *SVHN* (Netzer et al., 2011), *SYN* (Ganin & Lempitsky, 2015), and *USPS* (Denker et al., 1989)) is presented. We set $K = 15$ and $\beta = 1$ according to the best classification accuracy.

1) Number of MC samples (K). The classification accuracy on *Digits* (Volpi et al., 2018) with different K is shown in Fig. 8 (a). We notice that the average accuracy gradually increases from $K = 1$ to $K = 15$ and remains stable when $K = 20$.

2) Coefficient of constraint (β). The constraint is used to make adversarial domain augmentation satisfy the worst-case constraint. Results on *Digits* (Volpi et al., 2018) with different β is presented in Fig. 8 (b). As seen, the accuracy falls dramatically when $\beta = 10$, because large β may severely limit the domain transportation and create domain augmentations similar to the source.