A Bit More Bayesian: Domain-Invariant Learning with Uncertainty

Xiao, Z.; Shen, J.; Zhen, X.; Shao, L.; Snoek, C.G.M.

Publication date
2021

Document Version
Final published version

Published in
Proceedings of Machine Learning Research

License
Other

Citation for published version (APA):
A Bit More Bayesian: Domain-Invariant Learning with Uncertainty

Zehao Xiao 1 * Jiayi Shen 1 * Xiantong Zhen 1 2 Ling Shao 2 Cees G. M. Snoek 1

Abstract

Domain generalization is challenging due to the domain shift and the uncertainty caused by the inaccessibility of target domain data. In this paper, we address both challenges with a probabilistic framework based on variational Bayesian inference, by incorporating uncertainty into neural network weights. We couple domain invariance in a probabilistic formula with the variational Bayesian inference. This enables us to explore domain-invariant learning in a principled way. Specifically, we derive domain-invariant representations and classifiers, which are jointly established in a two-layer Bayesian neural network. We empirically demonstrate the effectiveness of our proposal on four widely used cross-domain visual recognition benchmarks. Ablation studies validate the synergistic benefits of our Bayesian treatment when jointly learning domain-invariant representations and classifiers for domain generalization. Further, our method consistently delivers state-of-the-art mean accuracy on all benchmarks.

1. Introduction

Learning to improve the generalization of deep neural networks to data out of their training distribution remains a fundamental yet challenging problem for machine learning (Wang et al., 2018; Bengio et al., 2019; Krueger et al., 2020). Domain generalization (Muandet et al., 2013) aims to train a model on several source domains and have it generalize well to unseen target domains. The main challenge stems from the large shift of distributions between the source and target domains, which is further complicated by the prediction uncertainty (Malinin & Gales, 2018) introduced by the inaccessibility to data from the target domains during training. Established approaches learn domain-invariant features by dedicated loss functions (Muandet et al., 2013; Li et al., 2018a) or specific architectures (Li et al., 2017; D’Innocente & Caputo, 2018). The state-of-the-art relies on advanced deep neural network backbones, known to degenerate when the test samples are out of the training data distribution (Nguyen et al., 2015; Ilse et al., 2019), due to their poorly calibrated behavior (Guo et al., 2017; Kristiadi et al., 2020).

Domain generalization is susceptible to uncertainty as the domain shift from the source to the target domain is unknown a priori. Hence, uncertainty should be taken into account during domain-invariant learning. As deep neural networks are commonly trained by maximum likelihood estimation, they fail to effectively capture model uncertainty. This tends to make the models overconfident in their predictions, especially on out-of-distribution data (Daxberger & Hernández-Lobato, 2019). As a possible solution, approximate Bayesian inference offers a natural framework to represent prediction uncertainty (Kristiadi et al., 2020; MacKay, 1992). It possesses better generalizability to out-of-distribution examples (Louizos & Welling, 2017) and provides an elegant formulation to transfer knowledge across different datasets (Nguyen et al., 2018). Moreover, the prediction uncertainty can be improved, even when Bayesian approximation is only applied to the last network layer (Kristiadi et al., 2020; Atanov et al., 2019). These properties make it appealing to introduce Bayesian learning into the challenging and, as of yet, unexplored scenario of domain generalization.

In this paper, we address domain generalization under a probabilistic framework1. To better explore domain-invariant learning, we introduce weight uncertainty to the model by leveraging variational Bayesian inference. To this end, we introduce the principle of domain invariance in a probabilistic formulation and incorporate it into the variational Bayesian inference framework. This enables us to explore domain invariance in a principled way to achieve domain-invariant feature representations and classifiers jointly. To better handle the domain shifts between seen and unseen domains, we explore our method under the meta-learning framework (Du et al., 2020; Balaji et al., 2018; Li & Malik, 2017). The meta-learning setting is utilized to

---

1Source code is publicly available at https://github.com/zzzx1224/A-Bit-More-Bayesian.git.
expose the model to domain shift and mimic the generalization process by randomly splitting source domains into several meta-source domains and a meta-target domain in each iteration. We evaluate our method on four widely-used benchmarks for cross-domain object classification. Our ablation studies demonstrate the benefit of domain-invariant learning in a probabilistic framework through its synergy with variational Bayesian inference, as well as the advantage of jointly learning domain-invariant feature extractors and classifiers for domain generalization. Our method achieves state-of-the-art mean accuracy on all four benchmarks.

2. Methodology

2.1. Preliminaries

In domain generalization, we have \( \mathcal{D} = \{ D_i \}_{i=1}^{\mathcal{D}} = \mathcal{S} \cup \mathcal{T} \) as a set of domains, where \( \mathcal{S} \) and \( \mathcal{T} \) denote the source and target domains, respectively. \( \mathcal{S} \) and \( \mathcal{T} \) do not have any overlap besides sharing the same label space. Data from the target domains \( \mathcal{T} \) is never seen during training. For each domain \( D_i \in \mathcal{D} \), we define a joint distribution \( p(x, y) \) on \( \mathcal{X} \times \mathcal{Y} \), where \( \mathcal{X} \) and \( \mathcal{Y} \) denote the input space and output space, respectively. We aim to learn a model \( f : \mathcal{X} \rightarrow \mathcal{Y} \) in the source domains \( \mathcal{S} \) that generalizes well to the target domains \( \mathcal{T} \).

We address domain generalization in a probabilistic framework of Bayesian inference by introducing weight uncertainty into neural networks. To be specific, we adopt variational Bayesian inference to learn a neural network that is assumed to be parameterized by weights \( \theta \) with a prior distribution \( p(\theta) \) and the posterior distribution \( p(\theta|x, y) \), where \( (x, y) \) are samples from the source domain \( \mathcal{S} \). To learn the model, we seek for a variational distribution \( q(\theta) \) to approximate \( p(\theta|x, y) \) by minimizing the Kullback-Leibler divergence \( \mathbb{D}_{KL}[q(\theta)||p(\theta|x, y)] \) between them. This amounts to minimizing the following objective:

\[
\mathcal{L}_{\text{Bayes}} = -\mathbb{E}_{q(\theta)}[\log p(y|x, \theta)] + \mathbb{D}_{KL}[q(\theta)||p(\theta)],
\]

which is also known as the negative value of the evidence lower bound (ELBO) (Blei et al., 2017). We learn the model on the source domain \( \mathcal{S} \) in the hope that it will generalize well to the target domain \( \mathcal{T} \).

Kristiadi et al. (2020) show that applying Bayesian approximation to the last layer of a neural network effectively captures model uncertainty. This is also appealing when dealing with uncertainty in domain generalization since applying Monte Carlo sampling to the full Bayesian neural network can be computationally infeasible. In this paper, to obtain a model that generalizes well across domains in an efficient way, we incorporate variational Bayesian approximation into the last two network layers to jointly establish domain-invariant feature representations and classifiers. More analyses are provided in the experiments and supplementary. To achieve better domain invariance, we introduce a domain-invariant principle in a probabilistic form under the Bayesian framework. We use \( \phi \) and \( \psi \) to denote the parameters of the feature extractor and classifier. We incorporate the domain-invariant principle into the inference of posteriors over \( \psi \) and \( \phi \), which results in our two-layer Bayesian network. Next we detail the Bayesian domain-invariant learning of the network and the objective for optimization.

2.2. Bayesian Domain-Invariant Learning

Existing domain generalizations try to achieve domain invariance by minimizing the distance between intra-class samples from different domains in the hope of reducing the domain gaps (Motiian et al., 2017). However, individual samples cannot be assumed to be representative of the distributions of samples from a certain domain. Therefore, it is preferable to minimize the distributional distance of intra-class samples from different domains, which can directly narrow the domain gap. A contrastive illustration is provided in Figure 1. Motivated by this, we propose to address domain generalization under the probabilistic modeling by

![Figure 1. Illustrative contrast between deterministic invariance (top) and probabilistic invariance (bottom), where colors indicate domains. The deterministic invariance tends to minimize the distance between two deterministic samples (a \( \rightarrow \) b). The samples come from their respective distributions, but with limited distributional awareness. This means that the samples cannot represent the complete distributions. Thus, the deterministic invariance can lead to small distances between samples, but large gaps between distributions, as shown in figure (b). In contrast, the probabilistic invariance directly minimizes the distance between different distributions (c \( \rightarrow \) d), which yields a better domain invariance as most of the samples in the distributions are taken into account.]

incorporating weight uncertainty into invariant learning in the variational Bayesian inference framework.

We first introduce the definition of domain invariance in a probabilistic form, which we adopt to learn domain-invariant feature representations and classifiers in a unified way. We define a continuous domain space \( \mathcal{D} \) containing a set of domains \( \{D_i\}_{i=1}^{\mathcal{D}} \) in \( \mathcal{D} \) and a domain-transform function \( g_\zeta(\cdot) \) with parameters \( \zeta \) in the domain space. \( \{D_i\}_{i=1}^{\mathcal{D}} \) are discrete samples in \( \mathcal{D} \). Function \( g_\zeta(\cdot) \) transforms samples \( x \) from a reference domain to a different domain \( D_\zeta \) with respect to \( \zeta \), where \( \zeta \sim q(\zeta) \), and different samples \( \zeta \) lead to different sample domains \( D_\zeta \sim \mathcal{D} \).

As a concrete example, consider the rotation domains in Rotated MNIST (Ghifary et al., 2015). \( \mathcal{D} \) is the rotation domain space, containing images rotated by continuous angles from 0 to \( 2 \pi \). A sample \( x \) from any domain in \( \mathcal{D} \) can be transferred into a different domain \( D_\zeta \) by:

\[
g_\zeta(x) = \begin{bmatrix} \cos(\zeta) & -\sin(\zeta) \\ \sin(\zeta) & \cos(\zeta) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \tag{2}
\]

where \( \zeta \sim \text{Uniform}(0, 2\pi) \). \( \zeta \) is the parameter of \( g_\zeta(\cdot) \) and different \( \zeta \) lead to different rotation angles of the original sample \( x \), which form another domain \( D_\zeta \) in \( \mathcal{D} \). In practice, transformations between domains are more complicated and the exact forms of \( g_\zeta(\cdot) \) and \( q(\zeta) \) are not explicitly known.

Based on the above assumptions about \( \mathcal{D} \), \( g_\zeta(\cdot) \) and \( q(\zeta) \), we introduce our definition of the probabilistic form of domain invariance as follows.

**Definition 2.1 (Domain Invariance)** Let \( x_i \) be a given sample from domain \( D_i \) in the domain space \( \mathcal{D} \), and \( x_i = g_\zeta(x_i) \) be a transformation of \( x_i \) in another domain \( D_\zeta \) from the same domain space, where \( \zeta \sim q(\zeta) \). \( p_\theta(y|x) \) denotes the output distribution of input \( x \) with model \( \theta \).

**Model \( \theta \) is domain-invariant in \( \mathcal{D} \) if**

\[
p_\theta(y_i|x_i) = p_\theta(y_i|x_\zeta), \quad \forall \zeta \sim q(\zeta). \tag{3}
\]

Here, we use \( y \) to represent the output from a neural layer with input \( x \), which can either be the prediction vector from the last layer or the feature vector from the last convolutional layer of a deep neural network.

To adopt the domain-invariant principle in the variational Bayesian inference framework, we reformulate (3) to an expectation form with respect to \( q_\zeta \), following Nalisnick & Smyth (2018):

\[
p_\theta(y_i|x_i) = E_{q_\zeta}[p_\theta(y_\zeta|x_\zeta)]. \tag{4}
\]

According to Definition 2.1, we use the Kullback-Leibler divergence between the two terms in (4), \( \mathbb{D}_{KL} [p_\theta(y_i|x_i)||E_{q_\zeta}[p_\theta(y_\zeta|x_\zeta)]] \), to quantify the domain invariance of the model, which will be zero when the model is domain invariant in the domain space \( \mathcal{D} \). To further facilitate the computation, we derive the upper bound of the KL divergence:

\[
\mathbb{D}_{KL} [p_\theta(y_i|x_i)||E_{q_\zeta}[p_\theta(y_\zeta|x_\zeta)]] \\
\leq E_{q_\zeta} \left[ \mathbb{D}_{KL} [p_\theta(y_i|x_i)||p_\theta(y_\zeta|x_\zeta)] \right], \tag{5}
\]

which can be approximated by Monte Carlo sampling as we usually have access to samples from different domains. The complete derivation of (5) is provided in the supplementary material.

We now adopt the probabilistic domain invariance principle in the variational Bayesian approximation of the last two layers with parameters of \( \phi \) and \( \psi \), respectively. The introduced weight uncertainty results in distributional representations \( p_\phi(z|x) \) and predictions \( p_\psi(y|z) \), based on which we derive domain-invariant learning.

**Invariant Classifier.** For the classifier with parameters \( \psi \) and input features \( z \), the probabilistic domain-invariant property in (5) is represented as \( E_{q_\zeta} \left[ \mathbb{D}_{KL} [p_\psi(y_i|z_i)||p_\psi(y_\zeta|z_\zeta)] \right] \). Under the Bayesian framework, the predictive distribution \( p_\psi(y|z) \) is obtained by taking the expectation over the distribution of parameter \( \theta \), i.e., \( p_\psi(y|z) = E_{q(\psi)}[p(y|z, \psi)] \). As the KL divergence is a convex function (Nalisnick & Smyth, 2018), we further extend it to the upper bound:

\[
E_{q_\zeta} \left[ \mathbb{D}_{KL} [p_\psi(y_i|z_i)||p_\psi(y_\zeta|z_\zeta)] \right] \\
= E_{q_\zeta} \left[ \mathbb{D}_{KL} [E_{q(\psi)}[p(y_i|z_i, \psi)]||E_{q(\psi)}[p(y_\zeta|z_\zeta, \psi)]] \right] \\
\leq E_{q_\zeta} \left[ E_{q(\psi)} \mathbb{D}_{KL} [p(y_i|z_i, \psi)||p(y_\zeta|z_\zeta, \psi)] \right], \tag{6}
\]

which is tractable with unbiased approximation by using Monte Carlo sampling.

In practice, we estimate the expectation over \( q(\zeta) \) in an empirical way. Specifically, in each iteration, we choose one domain from the source domains \( \mathcal{S} \) as the meta-target domain \( D_i \) and the rest are used as the meta-source domains \( \{D_j\}_{j=1}^{\mathcal{S}} \), where \( S = |\mathcal{S}| - 1 \). Then we use a batch of samples \( x_s \) from each meta-source domain in the same category as \( x_t \) to approximate \( x_\zeta = g_\zeta(x_t) \).

Thereby, the domain-invariant classifier is established by minimizing:

\[
\frac{1}{SN} \sum_{s=1}^{S} \sum_{i=1}^{N} E_{q(\psi)} \left[ \mathbb{D}_{KL} [p(y_i|z_i, \psi)||p(y_s^i|z_s^i, \psi)] \right], \tag{7}
\]

where \( \{z_s^i\}_{i=1}^{N} \) are representations of samples \( x_s \) from \( D_s \), which are in the same category as \( x_t \).
The variational Bayesian inference enables us to flexibly incorporate our domain invariant principle into different layers of the neural network. To enhance the domain invariance, we obtain domain-invariant representations by adopting the principle in the penultimate layer.

**Invariant Representations.** To obtain domain-invariant representations \( p(z|x) \) with feature extractor \( \phi \), we extend the quantification of our probabilistic domain invariance in (5) to \( E_q, KL[p(\phi(z|x_i)|p(\phi(z|x_c))] \). Based on the Bayesian layer, the probabilistic distribution of features \( p(\phi(z|x)) \) will be a factorized Gaussian distribution if the posterior of \( \phi \) is as well. We illustrate this as follows. Let \( \phi \) be the last Bayesian layer in the feature extractor with a factorized Gaussian posterior and \( x \) be the input feature of \( \phi \). The posterior of the activation \( z \) is also a factorized Gaussian (Kingma et al., 2015):

\[
q(\phi_{i,j}) \sim N(\mu_{i,j}, \sigma_{i,j}^2) \quad \forall \phi_{i,j} \in \phi
\Rightarrow p(z_i|x, \phi) \sim N(\gamma_j, \delta_j^2),
\]

\[
\gamma_j = \sum_{i=1}^N x_i \mu_{i,j}, \quad \text{and} \quad \delta_j^2 = \sum_{i=1}^N \sigma_{i,j}^2 \mu_{i,j},
\]

where \( z_j \) denotes the \( j \)-th element in \( z \), likewise for \( x_i \), and \( \phi_{i,j} \) denotes the element at position \((i, j)\) in \( \phi \). Thus, we assume the posterior of the last Bayesian layer in the feature extractor has a factorized Gaussian distribution. Then, it is easy to obtain the Bayesian domain invariance of the feature extractor.

In a similar way to (7), we estimate the expectation over \( q(z) \) empirically. Therefore, the domain-invariant representations are established by minimizing:

\[
\frac{1}{SN} \sum_{s=1}^S \sum_{i=1}^N \left[ KL[p(z_i|x_i, \phi)||p(z_i|x_i, \phi)] \right],
\]

where \( \{x_i\}_{i=1}^N \) are from \( D_s \), and denote the samples in the same category as \( x_i \). More details and an illustration of the Bayesian domain-invariant learning are provided in the supplementary material.

**2.3. Objective Function**

Having the Bayesian treatment for the last two layers with respect to parameters \( \phi \) and \( \psi \), the \( L_{Bayes} \) in (1) is instantiated as the objective with respect to \( \psi \) and \( \phi \) as follows:

\[
L_{Bayes} = -E_q[\psi] \left[ E_{q(\phi)}[\log p(y|x, \psi, \phi)] \right] + KL[q(\psi)||p(\psi)] + KL[q(\phi)||p(\phi)].
\]

The detailed derivation of (10) is provided in the supplementary material.

By integrating (7) and (9) into (10), we obtain the Bayesian domain-invariant learning objective as follows:

\[
L_{BIL} = \frac{1}{L} \sum_{i=1}^L \sum_{m=1}^M \left[ -\log p(y_t|x_t, \psi^{(t)}, \phi^{(m)}) \right] + \frac{1}{SN} \sum_{s=1}^S \sum_{i=1}^N \left[ KL[p(y_t|x_i, \psi^{(t)})||p(y_t^{s}|x_i^{s}, \psi^{(t)})] + \lambda_\psi KL[q(\psi)||p(\psi)] + \lambda_\phi KL[q(\phi)||p(\phi)] \right],
\]

where \( \lambda_\psi \) and \( \lambda_\phi \) are hyperparameters that control the domain-invariant terms. \( x_t \) and \( z_t \) denote the input and its feature from \( D_t \) and \( x_i^{s} \) and \( z_i^{s} \) are from \( D_s \). The posteriors are set to factorized Gaussian distributions, i.e., \( q(\psi) = N(\mu_\psi, \sigma_\psi^2) \) and \( q(\phi) = N(\mu_\phi, \sigma_\phi^2) \). We adopt the reparameterization trick to draw Monte Carlo samples (Kingma & Welling, 2014) as \( \psi^{(t)} = \mu_\psi + \epsilon^{(t)} \ast \sigma_\psi \)

When implementing our Bayesian invariant learning, to increase the flexibility of the prior distribution in our Bayesian layers, we place a scale mixture of two Gaussian distributions as the priors \( p(\psi) \) and \( p(\phi) \) (Blundell et al., 2015):

\[
\pi N(0, \sigma^2_1) + (1-\pi) N(0, \sigma^2_2),
\]

where \( \sigma_1, \sigma_2 \) are set according to (Blundell et al., 2015) and \( \pi \) is chosen by cross-validation. More experiments and analyses on the hyperparameters \( \lambda_\psi, \lambda_\phi \) and \( \pi \) are provided in the supplementary material.

**3. Related Work**

To address the domain shifts, domain adaptation (Saenko et al., 2010; Long et al., 2015; Wang et al., 2020) and domain generalization (Muandet et al., 2013; Li et al., 2017) are the two main settings. Domain adaptation has a key assumption that the data from the target domain is accessible during training, which is often invalid in realistic applications. By contrast, no target domain data is available during training in domain generalization, which introduces more prediction uncertainty into the problem and makes it more challenging. Our method is developed specifically for domain generalization.

One solution for domain generalization is to generate more source domain data to increase the probability of covering the data in the target domains (Shankar et al., 2018; Volpi et al., 2018). Shankar et al. (2018) augment the data by perturbing the input images with adversarial gradients generated by an auxiliary classifier. Qiao et al. (2020) propose an even more challenging scenario of domain generaliza-
Another solution for domain generalization involves learning domain-invariant features (D’Innocente & Caputo, 2018; Li et al., 2018b; 2017). Muandet et al. (2013) propose domain-invariant component analysis to learn invariant transformations by minimizing the dissimilarity across domains. Louizos et al. (2015) learn invariant representations by a variational auto-encoder (Kingma & Welling, 2014), introducing Bayesian inference into invariant feature learning. Both Dou et al. (2019) and Seo et al. (2019) achieve a similar goal by introducing two complementary losses and employing multiple normalizations. Li et al. (2019) propose an episodic training algorithm to obtain both a domain-invariant feature extractor and classifier. Zhao et al. (2020) propose entropy-regularization to learn domain-invariant features. Seo et al. (2019) and Zhou et al. (2021) design normalizations to combine different feature statistics and embed the samples into a domain-invariant feature space.

Meta-learning has also been considered for domain generalization. Li et al. (2018a) introduce a gradient-based method, i.e., model agnostic meta-learning (Finn et al., 2017), for domain generalization. Balaji et al. (2018) meta-learn a regularization function, making their model robust to domain shifts. Du et al. (2020) propose the meta-variational information bottleneck to learn domain-invariant representations through episodic training.

Gulrajani & Lopez-Paz (2020) find that with careful implementation, empirical risk minimization methods outperform many state-of-the-art models in domain generalization. They claim that model selection is non-trivial for domain generalization and algorithms for this task should specify their own model selection criteria, which is important for the completeness and comparability of the method. In this paper, we implement our method based on the pretrained ResNet-18, as done for our baseline and most methods we compare against in Section 4.

Motiian et al. (2017) previously considered representation alignment across domains in the same class. They propose a classification and contrastive semantic alignment loss based on the L2 distance between deterministic features. Different from them, we exploit Bayesian neural networks to learn domain-invariant representations by minimizing the distance between probabilistic distributions.

Bayesian neural networks have not yet been explored for domain generalization. Our method introduces variational Bayesian approximation to both the feature extractor and classifier of the neural network in conjunction with the newly introduced domain-invariant principle for domain generalization. The resultant Bayesian domain-invariant learning combines the representational power of deep neural networks and variational Bayesian inference.

4. Experiments

4.1. Datasets and Settings

We conduct our experiments on four widely used benchmarks for domain generalization and report the mean classification accuracy on target domains.

PACS2 (Li et al., 2017) consists of 9,991 images of seven classes from four domains – photo, art-painting, cartoon and sketch. We follow the “leave-one-out” protocol from (Li et al., 2017; 2018b; Carlucci et al., 2019), where the model is trained on any three of the four domains, which we call source domains, and tested on the last (target) domain.

Office-Home3 (Venkateswara et al., 2017) also has four domains: art, clipart, product and real-world. There are about 15,500 images of 65 categories for object recognition in office and home environments. We use the same experimental protocol as for PACS.

Rotated MNIST4 and Fashion-MNIST5 are introduced in (Piratla et al., 2020) for evaluating domain generalization. For fair comparison, we follow their recommended settings and randomly select a subset of 2,000 images from MNIST and 10,000 images from Fashion-MNIST, which are considered to have been rotated by 0°. The subset of images is then rotated by 15° through 75° in intervals of 15°, creating five source domains. The target domains are created by rotations of 0° and 90°. These datasets allow us to demonstrate the generalizability of our model by comparing its performance on in-distribution and out-of-distribution data.

Settings. For all four benchmarks, we employ a ResNet-18 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009) as the backbone. During training we use Adam optimization (Kingma & Ba, 2014) with a learning rate of 0.0001, and train for 10,000 iterations. In each iteration we choose one source domain as the meta-target domain. The batch size is 128. To fit the memory footprint, we choose a maximum number of samples per category and target domain to implement the domain-invariant learning, i.e., sixteen for

---

2https://domaingeneralization.github.io/
3https://www.hemanthdv.org/officeHomeDataset.html
4http://yann.lecun.com/exdb/mnist/
5https://www.kaggle.com/zalando-research/fashionmnist
The model with the highest validation set accuracy is shown in Figure 2 (d) and (b). We first conduct an ablation study on PACS to investigate the benefits of the Bayesian invariant classifier. Rows (b), (c), and (d) demonstrate the benefits of the Bayesian invariant classifier. Row (a) serves as our baseline model, which is a vanilla deep convolutional network without any Bayesian treatment or domain-invariant loss. The backbone is also a ResNet-18 pretrained on ImageNet. Rows (b), (c) and (d) show the performance with a Bayesian classifier, a deterministic domain-invariant classifier and our Bayesian-invariant classifier. By comparing (b) and (a), it is clear that the Bayesian treatment for the classifier improves the performance, especially in the “art-painting” and “sketch” domains. The deterministic invariant property, as shown in row (c), also benefits the performance. Nevertheless, our Bayesian invariant learning based on the probabilistic framework performs better. The results demonstrate that the Bayesian invariant learning enhances the robustness of Bayesian neural networks on out-of-distribution data and better leverages domain-invariance than the deterministic invariant model.

The three subfigures in the first row of Figure 2 also demonstrate the benefits of the Bayesian invariant classifier. The Bayesian treatment enlarges the inter-class distance in all domains, as shown in Figure 2 (b). The Bayesian layer incorporates weight uncertainty into the predictions and improves their diversity, which enhances the classification performance on the source domains as well as the generalization to the target domain. The deterministic invariant classifier tends to minimize the distance between samples with the same label. However, the effect is mainly on the source domain, without obvious impact on the target domain (pink samples), as shown in Figure 2 (c). By introducing uncertainty into both the classification and domain-invariant procedure, our Bayesian domain-invariant classifier further enlarges the inter-class distance in all domains and better generalizes the domain-invariant property to the target domain (compare Figure 2 (d) to (b) and (c)).

Benefits of Bayesian Invariant Classifier. In Table 1, rows (a) to (d) demonstrate the benefits of the Bayesian invariant classifier. Row (a) serves as our baseline model, which is a vanilla deep convolutional network without any Bayesian treatment or domain-invariant loss. The backbone is also a ResNet-18 pretrained on ImageNet. Rows (b), (c) and (d) show the performance with a Bayesian classifier, a deterministic domain-invariant classifier and our Bayesian-invariant classifier. By comparing (b) and (a), it is clear that the Bayesian treatment for the classifier improves the performance, especially in the “art-painting” and “sketch” domains. The deterministic invariant property, as shown in row (c), also benefits the performance. Nevertheless, our Bayesian invariant learning based on the probabilistic framework performs better. The results demonstrate that the Bayesian invariant learning enhances the robustness of Bayesian neural networks on out-of-distribution data and better leverages domain-invariance than the deterministic invariant model.

Benefits of Bayesian Invariant Feature Extractor. The benefit of our Bayesian domain-invariant principle for the feature extractor is demonstrated in rows (e), (f), and (g) of Table 1. Similar to the classifier, introducing Bayesian infe-
Figure 2. Illustration of the benefit of Bayesian domain-invariant learning by visualization of the feature representations. Colors denote domains, where the target domain “art-painting” is in violet, and shapes indicate classes. Our Bayesian domain-invariant learning (shown in the right column), especially when employed in both the classifier and feature extractor ((j)), achieves better results than the other cases shown in the ten subfigures, which correspond to the ten settings in Table 1 (identified by ID).

Benefits of Synergistic Bayesian Invariant Learning.
The last three rows of Table 1 show the performance when introducing Bayesian and invariant learning into both the classifier and feature extractor. Both the Bayesian learning (row (h)) and deterministic invariant learning (row (i)) in two layers perform better than introducing the corresponding properties into only one layer of the model (compared with (b), (c) and (e), (f)) on most domains, as well as in terms of mean performance. Overall, employing our Bayesian invariant layer in both the classifier and feature extractor achieves the best performance, as shown in row (j).

The last row in Figure 2 further demonstrates the benefits of synergistic Bayesian invariant learning. Incorporating the Bayesian classifier and the Bayesian feature extractor further maximizes the inter-class distance of all domains, as comparing (h) to (b) and (e). Introducing deterministic invariance into both the classifier and feature extractor also improves the domain-invariant property of features and the generalization to the target domain to some extent, as shown in subfigure (i). Moreover, by utilizing both the Bayesian domain-invariant classifier and Bayesian domain-invariant feature extractor, our method combines their benefits shown
Table 2. Performance of different priors on PACS. “Standard” denotes the standard Gaussian prior while “Mixture” denotes the scale mixture prior.

<table>
<thead>
<tr>
<th>Prior</th>
<th>Photo</th>
<th>Art-painting</th>
<th>Cartoon</th>
<th>Sketch</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>95.17 ±0.25</td>
<td>81.95 ±0.42</td>
<td>79.41 ±0.82</td>
<td>77.88 ±0.57</td>
<td>83.61 ±0.12</td>
</tr>
<tr>
<td>Mixture</td>
<td>95.97 ±0.24</td>
<td>83.92 ±0.71</td>
<td>81.61 ±0.59</td>
<td>80.31 ±0.91</td>
<td>85.45 ±0.24</td>
</tr>
</tbody>
</table>

Table 3. Effect of more Bayesian layers on PACS. The “Bayesian” and “Invariant” columns indicate whether the penultimate layer $\phi'$ in the feature extractor has a Bayesian and/or domain-invariant property. More Bayesian layers benefit the performance while excessive domain-invariant learning is harmful.

<table>
<thead>
<tr>
<th>$\phi'$</th>
<th>PACS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayesian</td>
<td>Invariant</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>✓</td>
<td>×</td>
</tr>
<tr>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 4. Comparison on PACS. Our method achieves the best performance on the “Cartoon” domain, is competitive on the other three domains and obtains the best overall mean accuracy.

<table>
<thead>
<tr>
<th>Photo</th>
<th>Art-painting</th>
<th>Cartoon</th>
<th>Sketch</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>92.85</td>
<td>75.12</td>
<td>77.44</td>
<td>75.72</td>
</tr>
<tr>
<td>Carlucci et al. (2019)</td>
<td>96.03</td>
<td>79.42</td>
<td>75.25</td>
<td>71.35</td>
</tr>
<tr>
<td>Dou et al. (2019)</td>
<td>94.99</td>
<td>80.29</td>
<td>77.17</td>
<td>71.69</td>
</tr>
<tr>
<td>Zhao et al. (2020)</td>
<td>96.65</td>
<td>80.70</td>
<td>76.40</td>
<td>71.77</td>
</tr>
<tr>
<td>Piratla et al. (2020)</td>
<td>94.10</td>
<td>78.90</td>
<td>75.80</td>
<td>76.70</td>
</tr>
<tr>
<td>Chattopadhyay et al. (2020)</td>
<td>93.35</td>
<td>76.90</td>
<td>80.38</td>
<td>75.21</td>
</tr>
<tr>
<td>Li et al. (2019)</td>
<td>93.90</td>
<td>82.10</td>
<td>77.00</td>
<td>73.00</td>
</tr>
<tr>
<td>Balaji et al. (2018)</td>
<td>95.50</td>
<td>83.70</td>
<td>77.20</td>
<td>70.30</td>
</tr>
<tr>
<td>Zhou et al. (2020)</td>
<td>96.20</td>
<td>83.30</td>
<td>78.20</td>
<td>73.60</td>
</tr>
<tr>
<td>Seo et al. (2019)</td>
<td>95.87</td>
<td>84.67</td>
<td>77.65</td>
<td>82.23</td>
</tr>
<tr>
<td>Huang et al. (2020)</td>
<td>95.99</td>
<td>83.43</td>
<td>80.31</td>
<td>80.85</td>
</tr>
<tr>
<td><strong>This paper</strong></td>
<td>95.97 ±0.24</td>
<td>83.92 ±0.71</td>
<td>81.61 ±0.59</td>
<td>80.31 ±0.91</td>
</tr>
</tbody>
</table>

Effect of Scale Mixture Priors. Instead of using an uninformative diagonal Gaussian prior, we adopt the scale mixture prior (Blundell et al., 2015). To show the effect of the prior, we conduct experiments with both the standard Gaussian prior and scale mixture Gaussian prior as in (12). The results are reported in Table 2. The scale mixture prior achieves better performance on all domains. We have also done an ablation on the scaling mixture prior by changing the value of $\pi$ in (12). The prior with $\pi=0.5$ achieves the best performance on the “cartoon” domain of PACS (Figure C.2 in supplementary), and thereby we use this value in all other experiments.

Effect of More Bayesian Layers. We also experiment with more Bayesian layers in the feature extractor, as shown in (d) and (g) and achieves the best performance, as shown in (j). This indicates the benefits of the synergy between Bayesian inference and probabilistic domain-invariant learning for domain generalization.

Effect of Scale Mixture Priors. Instead of using an uninformative diagonal Gaussian prior, we adopt the scale mixture prior (Blundell et al., 2015). To show the effect of the prior, we conduct experiments with both the standard Gaussian prior and scale mixture Gaussian prior as in (12). The results are reported in Table 2. The scale mixture prior achieves better performance on all domains. We have also done an ablation on the scaling mixture prior by changing the value of $\pi$ in (12). The prior with $\pi=0.5$ achieves the best performance on the “cartoon” domain of PACS (Figure C.2 in supplementary), and thereby we use this value in all other experiments.

Effect of More Bayesian Layers. We also experiment with more Bayesian layers in the feature extractor, as shown in Table 3. The settings of the model in the first row are the same as row (j) in Table 1. When introducing another Bayesian layer $\phi'$ without domain invariance into the model, as shown in the second row, the average performance improves slightly. However, if we introduce the Bayesian domain-invariant learning into $\phi'$ (third row), the overall performance deteriorates slightly. This may due to the loss of information in the features caused by the excessive use of domain-invariant learning. In addition, due to the Bayesian inference and Monte Carlo sampling, more Bayesian layers leads to higher memory usage and more computations (more detailed discussion in supplementary material). As such, we prefer to apply the Bayesian invariant learning only in the last feature extraction layer and the classifier.

State-of-the-Art Comparisons. We compare our method with the state-of-the-art on four datasets. For comprehensive comparison, we also include a vanilla deep convolutional ResNet-18 network as a baseline, without any Bayesian treatment, as done in row (a) of Table 1. The results for each
Table 5. Comparison on Office-Home. Our method achieves the best performance on the “Art” and “Clipart” domains, while being competitive on the “Product” and “Real” domains. Again we report the best overall mean accuracy.

<table>
<thead>
<tr>
<th></th>
<th>Art</th>
<th>Clipart</th>
<th>Product</th>
<th>Real</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>54.84</td>
<td>49.85</td>
<td>72.40</td>
<td>73.14</td>
<td>62.55</td>
</tr>
<tr>
<td>Carlucci et al. (2019)</td>
<td>53.04</td>
<td>47.51</td>
<td>71.47</td>
<td>72.79</td>
<td>61.20</td>
</tr>
<tr>
<td>Li et al. (2018b)</td>
<td>56.50</td>
<td>47.30</td>
<td>72.10</td>
<td>74.80</td>
<td>62.68</td>
</tr>
<tr>
<td>Seo et al. (2019)</td>
<td>59.37</td>
<td>45.70</td>
<td>71.84</td>
<td>74.68</td>
<td>62.90</td>
</tr>
<tr>
<td>Huang et al. (2020)</td>
<td>58.42</td>
<td>47.90</td>
<td>71.63</td>
<td>74.54</td>
<td>63.12</td>
</tr>
<tr>
<td>Zhou et al. (2020)</td>
<td>60.60</td>
<td>50.10</td>
<td>74.80</td>
<td>77.00</td>
<td>65.63</td>
</tr>
<tr>
<td>This paper</td>
<td>61.81±0.36</td>
<td>53.27±0.37</td>
<td>74.27±0.35</td>
<td>76.31±0.24</td>
<td>66.42±0.18</td>
</tr>
</tbody>
</table>

Table 6. Comparison on Rotated MNIST and Fashion-MNIST. In-distribution accuracy is evaluated on the test sets of MNIST and Fashion-MNIST with rotation angles of 15°, 30°, 45°, 60° and 75°, while out-of-distribution accuracy is evaluated on test sets with angles of 0° and 90°. Our method achieves best performance on both the in-distribution and out-of-distribution test sets.

<table>
<thead>
<tr>
<th></th>
<th>MNIST</th>
<th>Fashion-MNIST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In-distribution</td>
<td>Out-of-distribution</td>
</tr>
<tr>
<td>Baseline</td>
<td>98.4</td>
<td>93.5</td>
</tr>
<tr>
<td>Dou et al. (2019)</td>
<td>98.2</td>
<td>93.2</td>
</tr>
<tr>
<td>Piratla et al. (2020)</td>
<td>98.4</td>
<td>94.7</td>
</tr>
<tr>
<td>This paper</td>
<td><strong>99.0±0.02</strong></td>
<td><strong>96.5±0.08</strong></td>
</tr>
</tbody>
</table>

dataset are reported in Tables 4, 5 and 6.

On PACS, in Table 4, our method achieves the best mean accuracy. For each individual domain, we are competitive with the state-of-the-art and even exceed all other methods on the “cartoon” domain. On Office-Home, in Table 5, we again achieve the best mean accuracy. It is worth mentioning that on the most challenging “art” and “clipart” domains, we also deliver the highest accuracy, with a good improvement over previous methods. However, Zhou et al. (2020) and Seo et al. (2019) outperform the proposed model on some domains of PACS and Office-Home. In (Zhou et al., 2020), the source domains are augmented by a generator that synthesizes data from pseudo-novel domains, which often have similar characteristics with the source data. This pays off when the target data also has similar characteristics to the source domains, as the pseudo domains are more likely to cover the target domain, as can be seen for “product” and “real world” in Office-Home and “photo” in PACS. When the test domain is different from all of the training domains the performance suffers, e.g., “clipart” in Office-Home and “sketch” in PACS. We highlight that our method generates domain-invariant representations and classifiers, resulting in competitive results across all domains and overall. In addition, Seo et al. (2019) combine batch and instance normalization for domain generalization. This tactic is effective on PACS, but less so on Office-Home. We attribute this to the larger number of categories in Office-Home, where instance normalization is known to make features less discriminative with respect to object categories (Seo et al., 2019). In contrast, our Bayesian domain-invariant learning establishes domain-invariant features and predictions in a probabilistic form by introducing uncertainty into the model, resulting in good performance on both PACS and Office-Home.

On Rotated MNIST and Fashion-MNIST, following the experimental settings in Piratla et al. (2020), we evaluate our method on the in-distribution and out-of-distribution sets. As shown in Table 6, our method achieves the best performance on both sets of the two datasets. In particular, our method improves the classification performance on the out-of-distribution sets, demonstrating its strong generalizability to unseen domains, which is also consistent with the findings in Figure 2.

5. Conclusion

In this work, we propose a variational Bayesian learning framework for domain generalization. We introduce Bayesian neural networks into the model, which are able to better represent uncertainty and enhance the generalization to out-of-distribution data. To handle the domain shift between source and target domains, we propose a domain-invariant principle under the variational inference framework, which is incorporated by establishing a domain-invariant feature extractor and classifier. Our method combines the representational power of deep neural networks and uncertainty modeling ability of Bayesian learning, demonstrating effectiveness for domain generalization. Ablation studies further validate these benefits. Our Bayesian invariant learning sets a new state-of-the-art on four domain generalization benchmarks.
References


