Supplementary Materials for "A Bit More Bayesian: Domain-Invariant Learning with Uncertainty"

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

A. Derivation

A.1. Derivation of the Upper Bounds of Probabilistic Domain-invariant Learning

To implement $\mathbb{E}_{q_\zeta}\big[p_{\boldsymbol{\theta}}(\mathbf{y}_\zeta|\mathbf{x}_\zeta)\big]$ in a tractable way, we derive the upper bound in Section 2.2, which is achieved via Jensen's inequality:

$$\begin{aligned}\mathbb{D}_{\mathrm{KL}}\big[p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i)||\mathbb{E}_{q_\zeta}[p_{\boldsymbol{\theta}}(\mathbf{y}_\zeta|\mathbf{x}_\zeta)]\big] &= \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i)}\big[\log p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i) - \log \mathbb{E}_{q_\zeta}[p_{\boldsymbol{\theta}}(\mathbf{y}_\zeta|\mathbf{x}_\zeta)]\big] \\ &\leq \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i)}\big[\log p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i) - \mathbb{E}_{q_\zeta}[\log p_{\boldsymbol{\theta}}(\mathbf{y}_\zeta|\mathbf{x}_\zeta)]\big] \\ &= \mathbb{E}_{q_\zeta}\Big[\mathbb{D}_{\mathrm{KL}}\big[p_{\boldsymbol{\theta}}(\mathbf{y}_i|\mathbf{x}_i)||p_{\boldsymbol{\theta}}(\mathbf{y}_\zeta|\mathbf{x}_\zeta)\big]\Big].\end{aligned} \tag{1}$$

A.2. Derivation of Variational Bayesian Approximation for Feature Extractor (ϕ) and Classifier (ψ) layers

We consider the model with two Bayesian layers $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ as the last layer of the feature extractor and the classifier. The prior distribution of the model is $p(\boldsymbol{\phi}, \boldsymbol{\psi})$, and the true posterior distribution is $p(\boldsymbol{\phi}, \boldsymbol{\psi}|\mathbf{x}, \mathbf{y})$. Following the settings in Section 2.1, we need to learn a variational distribution $q(\boldsymbol{\phi}, \boldsymbol{\psi})$ to approximate the true posterior by minimizing the KL divergence from $q(\boldsymbol{\phi}, \boldsymbol{\psi})$ to $p(\boldsymbol{\phi}, \boldsymbol{\psi}|\mathbf{x}, \mathbf{y})$:

$$\boldsymbol{\phi}^*, \boldsymbol{\psi}^* = \arg\min_{\boldsymbol{\phi}, \boldsymbol{\psi}} \mathbb{D}_{\mathrm{KL}}\big[q(\boldsymbol{\phi}, \boldsymbol{\psi})||p(\boldsymbol{\phi}, \boldsymbol{\psi}|\mathbf{x}, \mathbf{y})\big]. \tag{2}$$

By applying the Bayesian rule $p(\boldsymbol{\phi}, \boldsymbol{\psi}|\mathbf{x}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}, \boldsymbol{\phi}, \boldsymbol{\psi})p(\boldsymbol{\phi}, \boldsymbol{\psi})$, the optimization is equivalent to minimizing:

$$\mathcal{L}_{\mathrm{Bayes}} = \int q(\boldsymbol{\phi}, \boldsymbol{\psi}) \log \frac{q(\boldsymbol{\phi}, \boldsymbol{\psi})}{p(\boldsymbol{\phi}, \boldsymbol{\psi})p(\mathbf{y}|\mathbf{x}, \boldsymbol{\phi}, \boldsymbol{\psi})} \mathrm{d}\boldsymbol{\phi}\mathrm{d}\boldsymbol{\psi} = \mathbb{D}_{\mathrm{KL}}\big[q(\boldsymbol{\phi}, \boldsymbol{\psi})||p(\boldsymbol{\phi}, \boldsymbol{\psi})\big] - \mathbb{E}_{q(\boldsymbol{\phi}, \boldsymbol{\psi})}\big[\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\phi}, \boldsymbol{\psi})\big]. \tag{3}$$

With $\boldsymbol{\phi}$ and $\boldsymbol{\psi}$ being independent,

$$\mathcal{L}_{\mathrm{Bayes}} = -\mathbb{E}_{q(\boldsymbol{\psi})}\mathbb{E}_{q(\boldsymbol{\phi})}[\log p(\mathbf{y}|\mathbf{x}, \boldsymbol{\psi}, \boldsymbol{\phi})] + \mathbb{D}_{\mathrm{KL}}[q(\boldsymbol{\psi})||p(\boldsymbol{\psi})] + \mathbb{D}_{\mathrm{KL}}[q(\boldsymbol{\phi})||p(\boldsymbol{\phi})]. \tag{4}$$

B. Details of Bayesian Domain-invariant Training

In domain generalization, the given domains $\mathcal{D}=\{D_i\}_{i=1}^{|\mathcal{D}|}$ are divided into the source domains $\mathcal{S}$ and the target domains $\mathcal{T}$. During training, as the data from $\mathcal{T}$ is never seen, only the source domains $\mathcal{S}$ are accessible. In each iteration, as shown in Figure B.1, we randomly choose one source domain from $\mathcal{S}$ as the meta-target domain D_t, and the rest of the source domains $\{D_s\}_{s=1}^{S}$ are treated as the meta-source domains, where $S=|\mathcal{S}|-1$. We randomly select a batch of samples $\mathbf{x}_t$ from D_t. We also select N samples $\{\mathbf{x}_s^i\}_{i=1}^N$, which are in the same category as $\mathbf{x}_t$, from each of the meta-source domains D_s. All these samples are sent to the Bayesian invariant feature extractor $\boldsymbol{\phi}$ after a ResNet18 backbone to get the representations $\mathbf{z}_t$ and $\{\mathbf{z}_s^i\}_{i=1}^N$, which are then sent to the Bayesian invariant classifier $\boldsymbol{\psi}$ to get the predictions $\mathbf{y}_t$ and $\{\mathbf{y}_s^i\}_{i=1}^N$. We obtain the Bayesian invariant objective function for the feature extractor $\mathcal{L}_I(\boldsymbol{\phi})$ by calculating the mean of the KL divergence of $p(\mathbf{z}_t|\mathbf{x}_t, \boldsymbol{\phi})$ and each $p(\mathbf{z}_s^i|\mathbf{x}_s^i, \boldsymbol{\phi})$ as (9) in the main paper. The Bayesian invariant objective function for feature classifier $\mathcal{L}_I(\boldsymbol{\psi})$ is calculated in a similar way on $p(\mathbf{y}_t|\mathbf{z}_t, \boldsymbol{\psi})$ and $\{p(\mathbf{y}_s^i|z_s^i, \boldsymbol{\psi})\}_{i=1}^N$ as (7) in the main paper. In addition to the Bayesian invariant objective functions, there are also a cross-entropy loss $\mathcal{L}_{CE}$ on $p(\mathbf{y}_t|\mathbf{x}_t, \boldsymbol{\psi}, \boldsymbol{\phi})$ and two Kullback-Leibler terms between the posteriors and priors of $\boldsymbol{\psi}$ and $\boldsymbol{\phi}$ as detailed in (11) in the main paper.

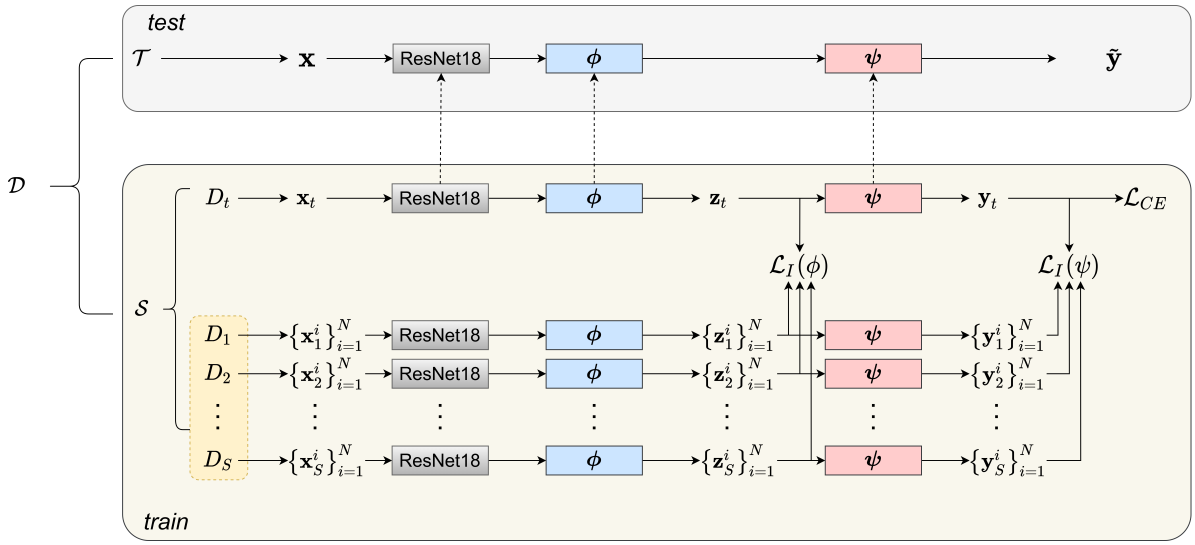

Figure B.1. Illustration of the training phase of our Bayesian domain-invariant learning. $\mathcal{S}$ denotes the source domains, $\mathcal{T}$ denotes the target domains, and $\mathcal{D} = \mathcal{S} \cup \mathcal{T}$. $\mathbf{x}$, $\mathbf{z}$ and $\mathbf{y}$ denote inputs, features and outputs of samples in each domain. $\mathcal{L}_I(\psi)$ and $\mathcal{L}_I(\phi)$ denote the domain-invariant objective functions for the classifier and feature representations as detailed in (7) and (9) in the main paper.

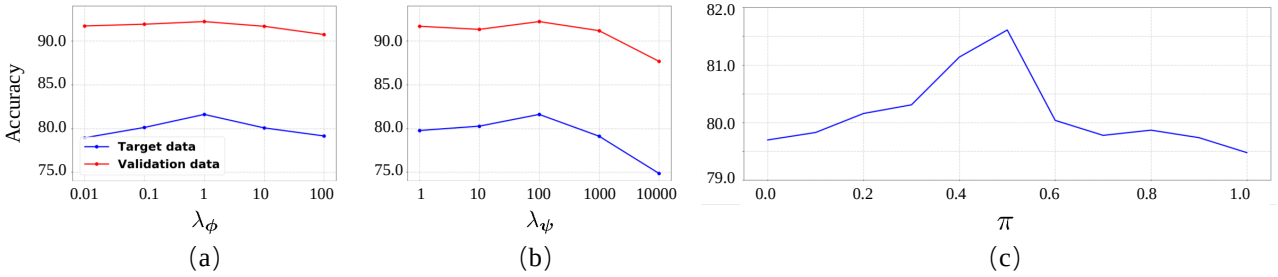

Figure C.2. Performance on "cartoon" domain in PACS with different hyperparameters λ_ϕ, λ_ψ and π. The red line denotes the accuracy on validation data while the blue line denotes accuracy on target data. The optimal value of λ_ϕ, λ_ψ and π are 1, 100 and 0.5.

C. Ablation Study for Hyperparamters

We also ablate the hyperparameters λ_ϕ, λ_ψ and π to show their effects. The experiments are conducted on PACS by using *cartoon* as the target domain. The results are shown in Figure C.2. Specifically, we produce Figure C.2 (a) by fixing λ_ψ as 100 and adjusting λ_ϕ, Figure C.2 (b) by fixing λ_ϕ as 1 and adjusting λ_ψ and Figure C.2 (c) by adjusting π while fixing other settings as in Section 4.1. λ_ϕ and λ_ψ balance the influence of the Bayesian learning and domain-invariant learning, and their optimal values are 1 and 100, respectively. If the values are too small, the model tends to overfit to source domains as the performance on target data drops more obviously than on validation data. By contrast, too large values of harm the overall performance of the model as there are obvious decrease of accuracy on both validation data and target data. Moreover, π balances the two components of the scale mixture prior of our Bayesian model. According to (Blundell et al., 2015), the two components cause a prior density with heavier tail while many weights tightly concentrate around zero. Both of them are important. The performance is the best when π is 0.5 according to Figure C.2 (c), which demonstrates the effectiveness of using two components in the scale mixture prior.

D. Additional Ablations on PACS

To further demonstrate the effectiveness of the Bayesian inference and Bayesian invariant learning, we conduct some more supplementary experiments with other settings on PACS as shown in Table D.1. For more comprehensive comparisons, we add four more settings with IDs (k), (l), (m), and (n) to the settings in Table 1.

Comparing (k) and (l) with (d), we find that when employing the Bayesian invariant classifier ψ in the model, introducing Bayesian or deterministic invariant learning into the last layer of the feature extractor ϕ both further improves the overall

Table D.1. More detailed ablation study on PACS. Compared to Table 1 we add four more settings with IDs (k), (l), (m) and (n). Bayesian inference benefits domain generalization. Our Bayesian invariant learning achieves better performance than the deterministic ones.

	Classifier ψ		**Feature extractor ϕ**		**PACS**				
ID	Bayesian	Invariant	Bayesian	Invariant	Photo	Art-painting	Cartoon	Sketch	*Mean*
(a)	×	×	×	×	92.85 ±0.21	75.12 ±0.48	77.44 ±0.26	75.72 ±0.47	80.28 ±0.42
(b)	✓	×	×	×	93.89 ±0.29	77.88 ±0.53	78.20 ±0.39	77.75 ±0.75	81.93 ±0.22
(c)	×	✓	×	×	93.95 ±0.51	80.03 ±0.72	78.03 ±0.77	77.83 ±0.52	82.46 ±0.67
(d)	✓	✓	×	×	95.21 ±0.26	81.25 ±0.76	80.67 ±0.73	79.31 ±0.94	84.11 ±0.39
(e)	×	×	✓	×	92.81 ±0.35	78.66 ±0.56	77.90 ±0.40	78.72 ±0.86	82.02 ±0.26
(f)	×	×	×	✓	94.17 ±0.35	79.75 ±0.68	79.51 ±0.98	78.31 ±1.11	82.94 ±0.53
(g)	×	×	✓	✓	95.15 ±0.26	80.96 ±0.69	79.57 ±0.85	79.15 ±0.98	83.71 ±0.65
(m)	✓	×	✓	✓	95.87 ±0.48	81.15 ±0.49	79.39 ±0.33	80.15 ±1.16	84.14 ±0.41
(n)	×	✓	✓	✓	95.39 ±0.12	82.32 ±0.37	80.27 ±0.58	79.61 ±0.93	84.40 ±0.45
(h)	✓	×	✓	×	93.83 ±0.19	82.13 ±0.41	79.18 ±0.48	79.03 ±0.78	83.54 ±0.34
(i)	×	✓	×	✓	94.12 ±0.22	80.52 ±0.61	80.39 ±0.81	78.53 ±0.95	83.39 ±0.52
(j)	✓	✓	✓	✓	**95.97** ±0.24	**83.92** ±0.71	**81.61** ±0.59	**80.31** ±0.91	**85.45** ±0.24

performance. Moreover, introducing Bayesian invariant learning into ϕ achieves the best improvements as shown in row (j). A similar phenomenon occurred on the classifier as comparing (m), (n) and (j) with (g). When there is a Bayesian invariant layer in the last layer of the feature extractor, introducing Bayesian learning or deterministic invariant learning both improves the performance, while Bayesian invariant learning achieves the best result. This further indicates the benefits of introducing uncertainty into the model and the invariant learning under the Bayesian framework.

E. Extra Visualizations

To further observe and analyze the benefits of the Bayesian invariant learning, we conduct more detailed visualizations of the features in Figure E.3 and Figure E.4. Visualizations in Figure E.3 show the features of all categories from the target domain only, while visualizations in Figure E.4 show features of only one category from all domains. Similar to Figure 2 in Section 4.2 of the main paper, the visualization is conducted on the PACS dataset and the target domain is "art-painting". The chosen category in Figure E.4 is "horse". We also visualize the features on the rotated MNIST and Fashion MNIST datasets in Figure E.5. The results further demonstrate the effectiveness of our Bayesian invariant learning on both in-distribution data and out-of-distribution data.

E.1. Visualizations on the Target Domain of PACS

Figure. E.3 provides a more intuitive observation of the benefits of Bayesian domain-invariant learning on the target domain.

- *Bayesian learning benefits classification on the target domain as shown in the "Bayesian" column of subfigures.* Comparing (b), (c) to (a), it is obvious that by introducing uncertainty into the model, both the Bayesian learning on the classifier and feature extractor enlarge the inter-class distance on the target domain. Samples from different categories tend to align on a thinner line, which makes them easier to classify. This phenomenon is more obvious when introducing Bayesian invariant learning into both the classifier and the feature extractor, as shown in (h).

- *Bayesian invariant learning is better than deterministic invariant learning.* Comparing the subfigures in the "Bayesian invariant" column to those in the "Deterministic invariant" column demonstrates the benefits of our Bayesian invariant learning. The Bayesian invariant classifier achieves larger inter-class distances than the deterministic invariant one, as shown in (d) and (c). The Bayesian invariant feature extractor pushes the features in all categories away from the center as shown in (g), which also results in larger inter-class distance. Moreover, employing the Bayesian invariant learning to both the classifier and the feature extractor conducts more obvious improvements, as comparing (j) to (i).

- *Bayesian invariant learning benefits generalization to the target domain by introducing uncertainty into both model and domain-invariant learning.* It is interesting to look at subfigures (b), (d), and (h). The visualizations in (d) are similar to (h), which also occurs in Figure 2 in the main paper. Compared to (b), (h) introduces more uncertainty by employing

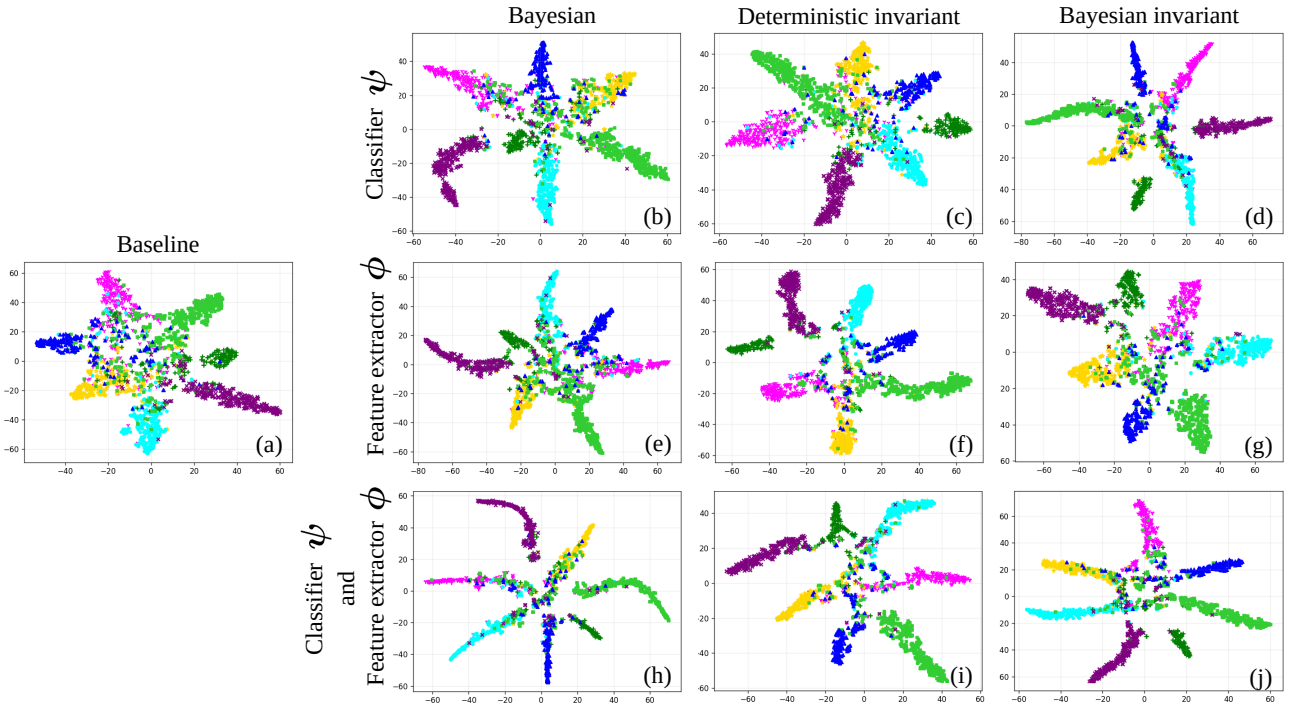

Figure E.3. Visualization of feature representations of the target domain. The subfigures have the same experimental settings as the experiments in Table 1 and Figure 2 in the main paper. To obtain more obvious contrast between the subfigures, we use different colors to denote different categories. The target domain is "art-painting", the same as in Figure 2. We obtain a similar conclusion to Section 4.2, where by introducing uncertainty into both the model and the domain-invariant learning, our Bayesian invariant learning achieves better performance and generalization to the target domain.

Bayesian inference in the feature extractor, while (d) applies the Bayesian invariant learning based on the Bayesian classifier. The similar observation in comparing (d) to (h) indicates that the benefits of our Bayesian invariant learning can be attributed to the uncertainty introduced into the domain-invariant learning. The objective function in (6) in the main paper also indicates this conclusion. Note that although both introduce uncertainty into the domain-invariant learning, visualization in (g) is different from (d). This is due to the uncertainty in the domain-invariant classifier and domain-invariant features having different effects on the feature space. As shown in (d) and (g), both the Bayesian invariant classifier and feature extractor enlarge the inter-class distance on the target domain, though from different directions.

E.2. Visualizations of Samples in One Class from Different Domains on PACS

Figure E.4 provides a deeper insight into the intra-class feature distributions of the same category from different domains.

- *Introducing uncertainty into the model gathers features from different domains to the same manifold.* By comparing (b), (e), and (h) to (a), we find that introducing Bayesian inference into the model tends to gather the features to a certain manifold. This is even more obvious with more Bayesian layers as shown in (h). The same phenomenon can be found in subfigure (d), where uncertainty is introduced in both the classifier and the domain-invariant learning. Gathering the features into a certain manifold is beneficial to domain generalization as it facilitates classification, both on the source domains and the target domain.

- *Bayesian invariant learning enables better generalization to the target domain than the deterministic counterpart.* The deterministic invariant learning minimizes the distance between deterministic samples from different source domains. That tends to be overfitting on the source domains. As shown in (c), (f), and (i), the samples from source domains are clustered well, while some samples from the target domains tend to be out of the cluster. There is even an obvious gap

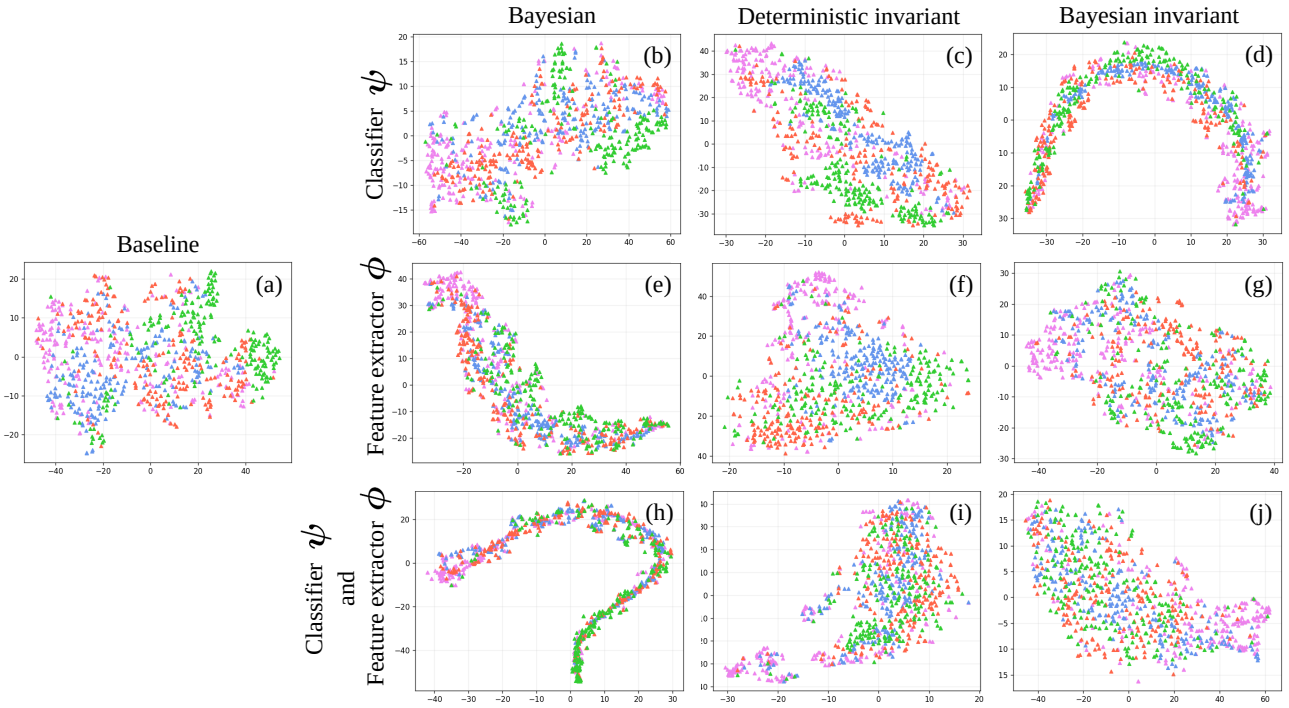

Figure E.4. Visualization of feature representations of one category. All samples are from the "horse" category with colors denoting different domains. The target domain is "art-painting" (violet). The "Bayesian" column shows Bayesian inference benefits domain generalization by gathering features from different domains to the same manifold. Comparing the subfigures in the "Bayesian invariant" column to those in the "Deterministic invariant" column indicates that our Bayesian invariant learning has better generalization to the target domain than the deterministic one.

when applying deterministic invariant learning on both classifier and feature extractor, as shown in (i). By contrast, our Bayesian invariant learning introduces uncertainty into domain-invariant learning. Thus, it achieves better mixture of features from the source domains and the target domain as shown in (d), (g), and (j). This indicates the Bayesian invariant learning enables better generalization to the target domain than the deterministic one.

E.3. Visualizations on Rotated MNIST and Fashion-MNIST

To further demonstrate the effectiveness of our Bayesian invariant learning, we also visualize the features on rotated MNIST and Fashion MNIST, as shown in Figure E.5. Different shapes denote different categories. Red samples denote features from the in-distribution set and blue samples denote features from the out-of-distribution set. Compared with the baseline, our method reduces the intra-class distance between samples from the in-distribution set as well as the out-of-distribution set, and clusters the out-of-distribution samples of the same categories better, especially in the rotated Fashion-MNIST dataset.

F. Computational Cost Analysis of Different Number of Bayesian Layers

To quantify the effect of the Bayesian framework on the computations and parameters of the model, we show the FLOPs and parameters for different numbers of Bayesian layers in Table F.2. The parameters of Bayesian layers are irrelevant to the batch size and the number of Monte Carlo samples. As the number of Bayesian layers increases, the extra parameters brought by the Bayesian layers grow smoothly, which will not cause much trouble. However, the extra FLOPs will have a significant increase with more Bayesian layers in the model. In addition, the FLOPs are related to the number of Monte Carlo samples and batch size. As shown in the third column, a bigger number of Monte Carlo samples leads to much higher FLOPs, especially with more Bayesian layers in the model. The extra FLOPs in the table are based on one image with size (224, 224, 3). When the batchsize becomes bigger, the FLOPs will also increase.

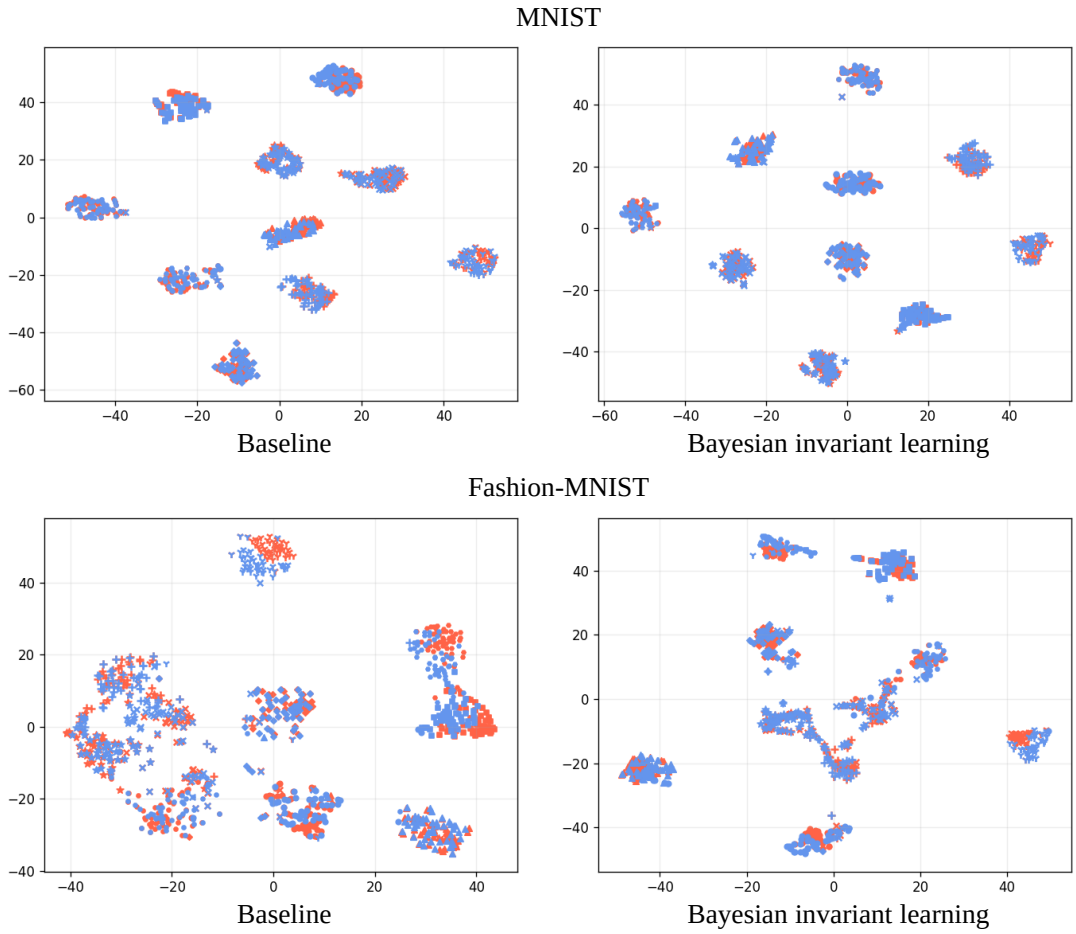

Figure E.5. Visualization of feature representations in rotated MNIST and rotated Fashion-MNIST datasets. Samples from the in-distribution and out-of-distribution sets are in red and blue, respectively. Different shapes denote different categories. Compared to the baseline, our Bayesian invariant learning achieves better performance on both the in-distribution and out-of-distribution sets in each dataset, and especially on the out-of-distribution set from the Fashion-MNIST benchmark.

Table F.2. Computational cost in FLOPs, parameters and GPU memory usage for different number of Bayesian layers in the model.

Number of Bayesian layers	Monte Carlo samples	Extra FLOPs (M)	Extra parameters (M)	Memory usage (G)
0	0	0	0	19.4
1	10	0.04	0.004	19.4
2	10	2.98	0.530	19.5
3	10	32.42	1.060	22.6
1	15	0.05	0.004	19.4
2	15	4.74	0.530	19.8
3	15	75.01	1.060	25.5
1	20	0.07	0.004	19.4
2	20	6.68	0.530	20.4
3	20	138.77	1.060	$>$ 31.7

To be more intuitive, we also show the GPU memory usage of different numbers of Bayesian layers. The experiments are conducted on the PACS dataset based on a single Tesla V100 GPU. The batch size is 128, and the number of samples per category and meta-source domain is 32. The GPU memory increases slightly when employing one or two Bayesian layers

Figure G.6. Some failure cases of our Bayesian invariant learning on PACS. The numbers associated with each image are the top two prediction probabilities of our method, with ground truth labels in red. Our method fails to make the correct predictions, but provides high probabilities for the true label, indicating the effectiveness of introducing uncertainty.

into the model, especially when the number of Monte Carlo samples is small. However, when introducing three Bayesian layers into the model, there will be a significant increase in GPU memory usage. When the number of Monte Carlo samples is bigger, such as 20, the model with three Bayesian layers will even run out of the GPU memory during training, as shown in the last row.

Based on Table F.2, we finally apply the Bayesian invariant learning only in the last feature extraction layer and the classifier. The batch size in our experiment is set to 128 and the number of Monte-Carlo sampling in each Bayesian layer is 10.

G. Failure Cases

Finally, we consider several failure cases of our method. As shown in Figure G.6, a complex scene makes it challenging to make correct predictions, e.g., (b), (d) in the "photo" domain and (d) in the "cartoon" domain. The failure cases in the "art painting" domain show the generalization performance of the method still needs further improvements, especially for samples with a very special and specific domain appearance. Our method also gets confused when samples have objects of different categories in the same image, e.g., (a) and (c) in the "photo" domain, or cluster the characteristics of different categories to the same sample, e.g., (a) and (b) in the "cartoon" domain and "sketch" domain. Although our Bayesian invariant learning fails to make the correct predictions in these challenging cases, it does provide reasonable probabilities for possible categories, which we attribute to the introduced model uncertainty and domain-invariant learning.

References

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. *ArXiv*, abs/1505.05424, 2015.