Distribution Shift Inversion for Out-of-Distribution Prediction
-Supplementary Material-

In this document, we provide additional materials that cannot fit into the main manuscript due to the page limit. First, we show experimental results on three more datasets. Extra transferred images are also presented. Next, we provide proof of the Theorem in the main text.

1. More Experimental Results and Implementation Details

1.1. Performance on ImageNet-R, ImageNet Sketch and CdSprites-5

The results on ImageNet-R [8], ImageNet Sketch [25] and CdSprites-5 [23] are provided in this section. We use [algorithm]* to indicate the use of DSI, the green cells to indicate our method improves the base method, and the **green cells with text in bold** to indicate our method improves the base method with non-overlapped confidence interval.

ImageNet-R is a variant of ImageNet [4] and contains 30,000 images from 200 classes with different styles. In OoD prediction, combined with a subset of the ImageNet, it is used as a single-training domain classification benchmark. In our

Dataset	CdSprites
ERM	47.46±0.16
ERM*	**89.39±0.52**
ANDMask [17]	47.56±0.16
ANDMask*	**89.88±0.18**
CAD [20]	47.62±0.32
CAD*	48.21±0.32
CondCAD [20]	47.53±1.40
CondCAD*	49.32±1.46
GroupDRO [21]	47.27±0.00
GroupDRO*	**89.91±0.05**
IB_ERM [1]	47.46±0.16
IB_ERM*	**89.39±0.52**
IB_IRM [1]	47.36±0.08
IB_IRM*	**89.55±0.52**
Mixup [26]	47.69±0.33
Mixup*	**89.78±0.30**
SANDMask [22]	47.72±0.28
SANDMask*	**89.55±0.44**
SelfReg [11]	47.36±0.08
SelfReg*	**89.52±0.52**
Average Gain	33.95±16.38

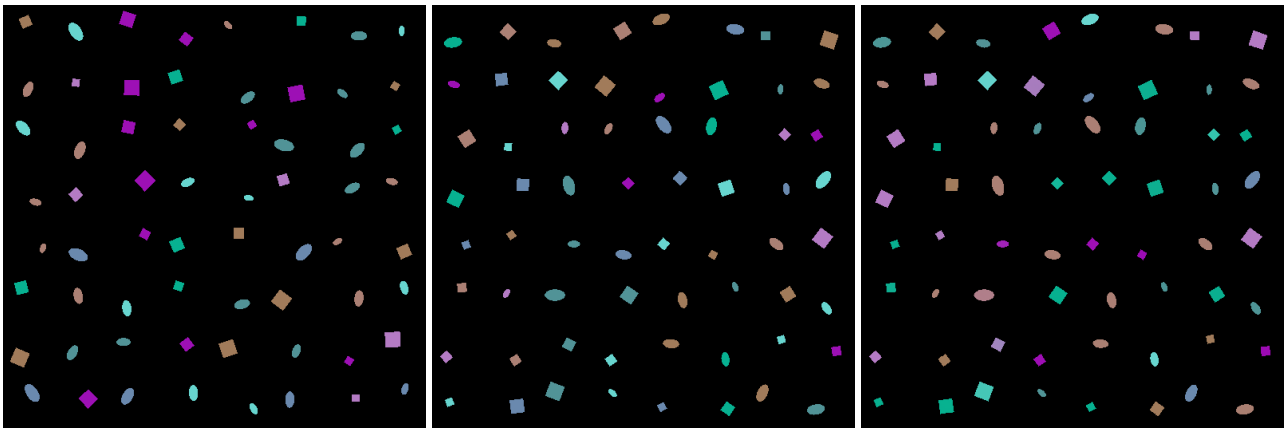
Table 1. The average accuracy ± the standard deviation of base algorithms w/o our method on CdSprites. The performance is generally boosted when our method is plugged in, whichever base algorithm used.

Dataset	ImageNet-R	ImageNet Sketch
ERM	36.19±1.66	20.14±1.20
ERM*	37.28±0.14	21.08±0.15
CORAL	36.86±1.65	18.11±1.09
CORAL*	38.02±0.06	**19.86±0.18**
RSC	35.58±1.53	18.44±0.96
RSC*	**37.80±0.12**	**20.36±0.44**
SagNet	35.26±0.95	19.50±1.15
SagNet*	**37.49±0.67**	20.41±0.16
Average Gain	1.68±0.55	1.38±0.46

Table 2. The average accuracy ± the standard deviation of base algorithms w/o our method on ImageNet-R and ImageNet Sketch datasets with single training domain setting. The performance is generally boosted when our method is plugged in, whichever base algorithm used.

Condition	**Parameter**	**Value**
DDPM	learning rate	$1e-4$
	batch size	32
	diffusion step	$4,000$
Stable Diffusion	learning rate	$1e-4$
	batch size	32
	diffusion step	$1,000$

Table 3. Diffusion Model Hyperparameters.



(a) Training Samples (b) Testing Samples (c) Transformed Samples

Figure 1. Samples from CdSprites-5 dataset. Testing and transformed samples are in one-to-one correspondence.

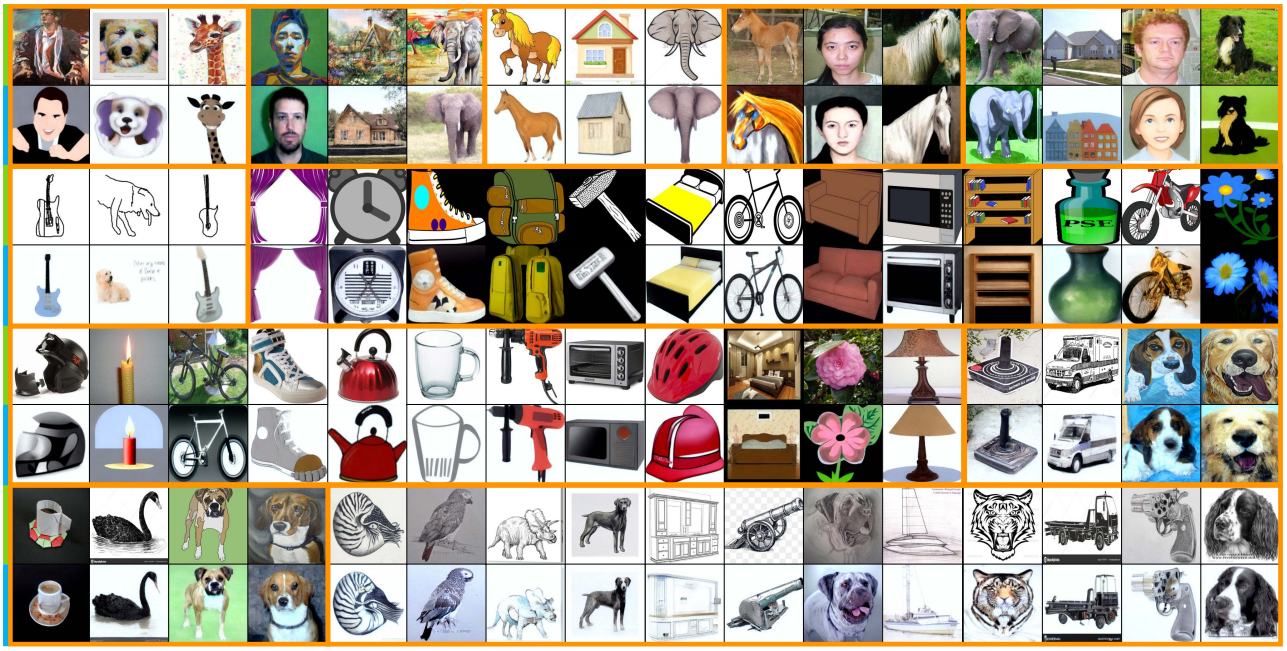
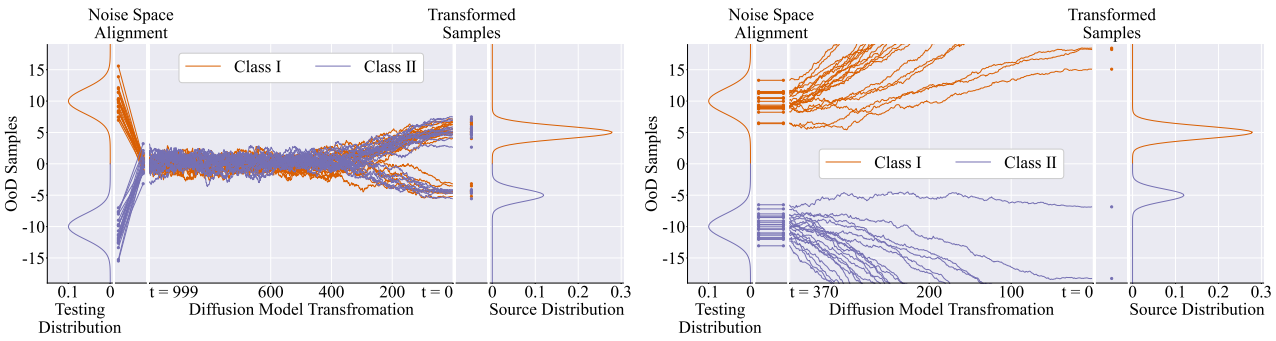


Figure 2. Transformed OoD samples. **Odd rows** show the original OoD images, and **even rows** show their transformation results to the source distribution. Samples in the same **box** are form the same source-target pair.

experiment, we select the images, whose labels appear in the ImageNet-R, from the training set of ImageNet as the training set and use ImageNet-R as the testing set. The results is shown in Tab. 2. Our method improves the base method by 1.68% on average.

ImageNet Sketch is another variant of ImageNet and contains 50,000 sketch images collected by Google search engine, which leads to 50 sketch images for each of the ImageNet classes. In the experiments, the original training set of ImageNet is used as the training domain, and ImageNet Sketch is used as the testing domain. The results is shown in Tab. 2. Our method improves the base method by 1.38% on average. This shows that our method is effective for dataset with large number of classes.

CdSprites-5 is a variant of DSprites [15]. The original DSprites contains white shapes with different scales, rotations, and positions. CdSprites-5 construct a binary classification problem by selecting 2 shapes from DSprites. There are 5 training domains, 1 validation domain and 1 testing domain in CdSprites-5. Instead of using the white shapes, the author color the



(a) Total Alignment

(b) No Alignment, $s = 370$

Figure 3. With or without noise space alignment in the 1-D distribution transformation example.

shape. For different training domain, different color pairs are used. In the validation and testing domain, all colors appear in the training domains are used. Furthermore, spurious correlation is injected to the training set by makes the colors strongly correlated to the shapes. While in the validation and testing set, the colors and the shapes are uncorrelated. The results is shown in Tab. 1. Our method achieves significant improvement when combined with base methods. This shows that our method is also able to address the distribution shift induced by covariate shift and spurious correlation.

1.2. Transformed Samples

Fig. 1 shows the training, testing and the transformed samples on the CdSprites dataset. Each testing sample and each transformed sample are corresponded. As shown, in the figure, for almost all sprites, the transformation does not change the shape information, but correct the color information.

Fig. 2 also provide extra transformed sample together with the corresponding OoD samples on PACS, OfficeHome, ImageNet-R and ImageNet Sketch.

1.3. Diffusion Model Implementation

For CDSprites-5, a DDPM model is trained from scratch with the default structure in [10]. For other datasets, we fine-tune the U-net structure of the stable diffusion [19][1] pretrained on the laion-aesthetics v2 5+. Other important hyperparameters are listed in Tab. 3. AdamW [14] optimizer is used for all the Diffusion Models. After training, the diffusion step is down-scaled to 250 for both types of diffusion model by a linear schedule.

1.4. OoD Algorithms Implementation

For CDSprites, a 7-layer Convolutional Neural Network with ReLU activation, BatchNorm and residual shortcut is used. For other datasets, EfficientNet is used. All of the algorithms are optimized by Adam [12]. To be consistent with previous works, we reuse the hyperparameter search regions in [7] and the subsequent updates of its implementation. For completeness, here, we also list the hyperparameter search regions in Tab. 4.

1.5. No Guidance

Though guidance is a frequently and widely used guarantee for image generation quality when implementing Diffusion Model, such as the classifier-based [5], the CLIP-based [2, 16], the classifier-free [18], and the reference image-based [2, 3, 6] guidance. We free DSI from using this type of technique based on the following reasons. First, in the OoD prediction task, though can be guessed in a self-supervised learning style, no explicit prompts or labels are available before the OoD prediction is made. This makes the first three guidance techniques inapplicable. Second, the reference image-based guidance is task-specific and strictly preserves certain low-level components of the reference, for example, the low-frequency components [3, 6] or parts of the reference [2]. While taking the OoD sample itself as the reference image is possible, there are no low-level components that have a guarantee on the OoD prediction and should be preserved.

[1] named sd-v1-4.ckpt in the official github repository.

Condition	Parameter	Region
Common	weight decay	$10^{\text{Uniform}(-6,-2)}$
	generator weight decay	$10^{\text{Uniform}(-6,-2)}$
EfficientNet (1000class ImageNet)	learning rate	$10^{\text{Uniform}(-3.5,-2.5)}$
	batch size	$2^{\text{Uniform}(8.5,9.5)}$
	generator learning rate	$10^{\text{Uniform}(-5,-3.5)}$
	discriminator learning rate	$10^{\text{Uniform}(-5,-3.5)}$
EfficientNet (Others)	learning rate	$10^{\text{Uniform}(-5,-3.5)}$
	batch size	$2^{\text{Uniform}(3,5.5)}$
	generator learning rate	$10^{\text{Uniform}(-5,-3.5)}$
	discriminator learning rate	$10^{\text{Uniform}(-5,-3.5)}$
7layer CNN	learning rate	$10^{\text{Uniform}(-4.5,-3.5)}$
	batch size	$2^{\text{Uniform}(3,9)}$
	generator learning rate	$10^{\text{Uniform}(-4.5,-2.5)}$
	discriminator learning rate	$10^{\text{Uniform}(-4.5,-2.5)}$
ANDMask	tau	$\text{Uniform}(0.5, 1)$
CAD/ConCAD	lambda	$\text{Choice}(1e-4, 1e-3, 1e-2, 1e-1, 1, 1e1, 1e2)$
	temperature	$\text{Choice}(0.05, 0.1)$
GroupDRO	eta	$10^{\text{Uniform}(-1,1)}$
IB ERM	lambda	$10^{\text{Uniform}(-1,5)}$
	penalty anneal iter	$int(10^{\text{Uniform}(0,4)})$
IB IRM	lambda	$10^{\text{Uniform}(-1,5)}$
	penalty anneal iter	$int(10^{\text{Uniform}(0,4)})$
	irm lambda	$10^{\text{Uniform}(-1,5)}$
	irm penalty anneal iter	$int(10^{\text{Uniform}(0,4)})$
Mixup	alpha	$10^{\text{Uniform}(0,4)}$
SANDMask	tau	$\text{Uniform}(0.5, 1)$
	k	$10^{\text{Uniform}(-3,5)}$
CORAL	gamma	$10^{\text{Uniform}(-1,1)}$
RSC	rsc f drop factor	$\text{Uniform}(0, 0.5)$
	rsc b drop factor	$\text{Uniform}(0, 0.5)$
SagNet	sag w adv	$10^{\text{Uniform}(-2,1)}$

Table 4. OoD Algorithm Hyperparameters.

1.6. Different Confidence Scores

We evaluate our method with another two confidence metrics: Maximal Likelihood [13] and KL-Divergence [9]. The results are shown in Tab. 5. Despite the confidence score used, our method can improve the baseline method on average. The results also indicate the importance of the choice of confidence score. With an improper confidence score, the performance of our method degenerates.

1.7. Experiments Under Domainbed

Experimental results using Domainbed implementation are shown in Tab. 6. The main difference between the experiment in this subsection and the experiments in the main text and above is the architecture of the predictor. Our method improves the baseline method on average.

Confidence Score	Algorithm	A	C	P	S	Average
Maximal Likelihood	ERM	85.64±1.17	80.44±0.74	96.97±0.49	77.73±3.97	85.20±1.60
	ERM*	88.96±0.98	85.06±1.24	97.56±0.21	85.42±3.36	89.25±1.44
	SelfReg	84.83±2.44	78.32±3.27	95.48±0.62	78.65±4.55	84.32±2.72
	SelfReg*	87.79±1.91	82.65±2.49	96.13±0.51	85.22±2.80	87.95±1.93
KL-Divergence	ERM	85.64±1.17	80.44±0.74	96.97±0.49	77.73±3.97	85.20±1.60
	ERM*	85.74±0.39	81.35±1.60	96.35±0.53	80.60±4.58	86.01±1.77
	SelfReg	84.83±2.44	78.32±3.27	95.48±0.62	78.65±4.55	84.32±2.72
	SelfReg*	83.79±3.45	79.26±2.97	94.92±0.55	80.63±4.07	84.65±2.76

Table 5. The average accuracy ± the standard deviation of base algorithms w/o our method on PACS using different confidence scores.

Confidence Score	Algorithm	A	C	P	S	Average
Training-Set Validation	ERM	84.41±3.01	80.44±0.74	96.81±0.53	79.82±1.80	85.69±1.52
	ERM*	84.64±2.42	81.38±1.60	96.84±0.54	82.52±1.04	86.91±1.40
	SelfReg	84.83±2.44	78.55±4.80	95.61±0.50	79.00±1.21	84.39±2.24
	SelfReg*	84.97±2.51	80.86±3.69	95.61±0.50	80.40±0.60	85.62±1.83
Testing-Set Validation	ERM	85.64±1.17	80.44±0.74	96.42±0.90	77.73±3.97	85.06±1.70
	ERM*	85.74±0.39	81.38±1.60	96.88±0.44	80.60±4.58	86.15±1.75
	SelfReg	84.83±2.44	78.32±3.27	95.64±0.40	78.65±4.55	84.36±2.67
	SelfReg*	84.97±2.51	79.30±3.01	95.67±0.33	80.53±4.29	85.12±2.54

Table 6. The average accuracy ± the standard deviation of base algorithms w/o our method on PACS using Domainbed implementation.

1.8. Cost Analyses and Practical Suggestions

Inference. Given starting time series $\{s_l\}_{l=0}^{L}$ (defined in Alg. 1) and M source domains, during inference, the neural network of the base method and each diffusion model forward at most $M \times L + 1$ times and $\sum_{l=0}^{L} s_l$ times, respectively. With diffusion acceleration methods, our method can use smaller s_l's and L to reduce the inference cost.

Training. The training cost of our method is mainly influenced by the number of source domains and the number of samples required. Our method uses one diffusion model per source domain and one classifier for all source domains. As the number of source-domain grows, the network training cost increases linearly. To reduce the number of diffusion models, one possible way is to group similar domains and train a shared diffusion model for each group. Our method uses pre-trained diffusion models and then fine-tunes them to improve sample efficiency. In our experiments, with about $1k$ samples for each source domain, the fine-tuned diffusion models can have desired performance. Few-shot fine-tuning techniques can further reduce the training sample consumption.

2. Further Discussion on 1-D UDT

We dig deeper into our 1-D UDT example. to show that the noise space alignment is crucial. The results are plotted in Fig. 3. When the OoD samples are entirely aligned to the noise, label information is lost completely.(Fig. 3a) Without using noise space alignment, the OoD sample are also Out-of-Distribution with respect to the Diffusion Model. When the original OoD samples are directly fed into the Diffusion Model, the evolution behavior is uncontrollable.(Fig. 3b)

3. Proof of Theorem 1

Theorem 1. *Given a diffusion model trained on the source distribution $p(\boldsymbol{x})$,let p_t denote the distribution at time t in the forward transformation, let $\bar{p}(\boldsymbol{x})$ denote the output distribution when the input of the backward process is standard Gaussian noise $\boldsymbol{\epsilon}$ whose distribution is denoted by $\rho(\boldsymbol{x})$, let $\omega(\boldsymbol{x})$ denote the output distribution when the input of backward process is a convex combination $\hat{\boldsymbol{X}} = (1-\alpha)\boldsymbol{X}' + \alpha\boldsymbol{\epsilon}$, where random variable $\boldsymbol{X}'$ is sampled following the target distribution $q(\boldsymbol{x})$ and $\alpha \in (0,1)$. Under some regularity conditions listed below, we have*

$$KL(p||\omega) \leqslant \mathcal{J}_{SM} + KL(p_T||\rho) + \mathcal{F}(\alpha) \tag{1}$$

To prove Theorem 1, we make the following assumptions. Assumptions (i) to (xii) are required for implementing Theorem 1 in [24]. Specifically, assumptions (i) & (ii) require the source distribution and noise distribution to be differentiable and have finite variance, assumptions (iii)-(iv) require f or the difference of f in Eq. (2) to be bounded corresponding to its input or the difference of its inputs, assumption (iv) requires g in Eq. (2) to be non-zero, assumption (vi) requires p_t (defined in Section 3) and its derivative to be bounded. assumptions (vii)-(viii) require the score function of p_t or the difference of it to be bounded corresponding to its input or the difference of its inputs, assumptions (ix)-(x) require that the estimated score function or the difference of it to be bounded corresponding to its input or the difference of its inputs, assumption (x) requires the estimation error is not infinitely large, assumption (xii) requires the value of $\boldsymbol{X}$ to be bounded, *e.g.*, bounding the values to $[0, 255]$ for images. Assumption (xiii) is used to constrain that $\mathcal{F}$ has a finite first-order derivative.

(i) $p(\boldsymbol{X}) \in \mathcal{C}^2$ and $\mathbb{E}_{\boldsymbol{X}\sim p}\big[\, \|\boldsymbol{X}\|_2^2 \,\big] < \infty$.

(ii) $\omega_T(\boldsymbol{X}) \in \mathcal{C}^2$ and $\mathbb{E}_{\boldsymbol{X}\sim \omega_T}\big[\, \|\boldsymbol{X}\|_2^2 \,\big] < \infty$.

(iii) $\forall t \in [0,T] : f(\cdot,t) \in \mathcal{C}^1, \exists C > 0\ \forall \boldsymbol{X} \in \mathbb{R}^D, t \in [0,T] : \|f(\boldsymbol{X},t)\|_2 \leqslant C(1 + \|\boldsymbol{X}\|_2)$.

(iv) $\exists C > 0, \forall \boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^D : \|f(\boldsymbol{X},t) - f(\boldsymbol{Y},t)\|_2 \leqslant C\,\|\boldsymbol{X} - \boldsymbol{Y}\|_2$.

(v) $g \in \mathcal{C}$ and $\forall t \in [0,T], |g(t)| > 0$.

(vi) For any open bounded set $\mathcal{O}$, $\int_0^T \int_{\mathcal{O}} \|p_t(\boldsymbol{X})\|_2^2 + Dg(t)^2\, \|\nabla_{\boldsymbol{X}} p_t(\boldsymbol{X})\|_2^2 \,\mathrm{d}\boldsymbol{X}\,\mathrm{d}t < \infty$.

(vii) $\exists C > 0\ \forall \boldsymbol{X} \in \mathbb{R}^D, t \in [0,T] : \|\nabla_{\boldsymbol{X}} \log p_t(\boldsymbol{X})\|_2 \leqslant C(1 + \|\boldsymbol{X}\|_2)$.

(viii) $\exists C > 0, \forall \boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^D : \|\nabla_{\boldsymbol{X}} \log p_t(\boldsymbol{X}) - \nabla_{\boldsymbol{Y}} \log p_t(\boldsymbol{Y})\|_2 \leqslant C\,\|\boldsymbol{X} - \boldsymbol{Y}\|_2$.

(ix) $\exists C > 0\ \forall \boldsymbol{X} \in \mathbb{R}^D, t \in [0,T] : \|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{X},t)\|_2 \leqslant C(1 + \|\boldsymbol{X}\|_2)$.

(x) $\exists C > 0, \forall \boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^D : \|\boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{X},t) - \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{Y},t)\|_2 \leqslant C\,\|\boldsymbol{X} - \boldsymbol{Y}\|_2$.

(xi) Novikov's condition:
$\mathbb{E}\Big[\, \exp\Big(\frac{1}{2}\int_0^T \|\nabla_{\boldsymbol{X}} \log p_t(\boldsymbol{X}) - \boldsymbol{s}_{\boldsymbol{\theta}}(\boldsymbol{X},t)\|_2^2 \,\mathrm{d}t\Big)\Big] < \infty$.

(xii) $\forall t \in [0,T]\ \exists k > 0 : p_t(\boldsymbol{X}) = O(e^{-\|\boldsymbol{X}\|_2^k})$ as $\|\boldsymbol{X}\|_2 \to \infty$.

(xiii) $\exists C_1 > 0$ and $C_2 > 0 : |\mathbb{E}_{\boldsymbol{X}\sim q}\big[\boldsymbol{X}\big]| < C_1$ and $|\mathbb{E}_{\boldsymbol{X}\sim p_T}\big[\boldsymbol{X}\big]| < C_2$.

Proof. The proof is composed of two parts. First, we bounded the KL-divergence between the p_T and the convex combination $\hat{X}$. Second, we bound the KL-divergence between the generated distribution ω and the source distribution p and show the convergence of $\mathcal{F}(\alpha)$.

Part 1. The distribution of $\hat{X}$ is in the form of the following convolution,

$$\omega_T(\boldsymbol{x}) = \int \frac{1}{\alpha(1-\alpha)}\rho(\frac{\boldsymbol{x}-\boldsymbol{\tau}}{\alpha})q(\frac{\boldsymbol{\tau}}{1-\alpha})d\boldsymbol{\tau} \tag{2a}$$

$$= \int \frac{1}{\alpha(1-\alpha)}\frac{1}{\sqrt{2\pi}}e^{-\frac{||\boldsymbol{x}-\boldsymbol{\tau}||_2^2}{2\alpha^2}}q(\frac{\boldsymbol{\tau}}{1-\alpha})d\boldsymbol{\tau} \tag{2b}$$

$$= \int \frac{1}{\alpha(1-\alpha)}\frac{1}{\sqrt{2\pi}}e^{-\frac{||\boldsymbol{x}||_2^2}{2\alpha^2}}e^{-(\frac{||\boldsymbol{\tau}||_2^2}{2\alpha^2}-\frac{\boldsymbol{x}^T\boldsymbol{\tau}}{\alpha^2})}q(\frac{\boldsymbol{\tau}}{1-\alpha})d\boldsymbol{\tau} \tag{2c}$$

$$= \frac{1}{\alpha}\rho(\frac{\boldsymbol{x}}{\alpha})\int \frac{1}{1-\alpha}e^{-(\frac{||\boldsymbol{\tau}||_2^2}{2\alpha^2}-\frac{\boldsymbol{x}^T\boldsymbol{\tau}}{\alpha^2})}q(\frac{\boldsymbol{\tau}}{1-\alpha})d\boldsymbol{\tau} \tag{2d}$$

$$= \frac{1}{\alpha}\rho(\frac{\boldsymbol{x}}{\alpha})\int e^{-\frac{1}{2\alpha^2}[(1-\alpha)^2||\boldsymbol{\nu}||_2^2-2(1-\alpha)\boldsymbol{x}^T\boldsymbol{\nu}]}q(\boldsymbol{\nu})d\boldsymbol{\nu} \tag{2e}$$

$$= \rho(\boldsymbol{x})\int e^{-\frac{1}{2\alpha^2}[(1-\alpha)^2||\boldsymbol{\nu}||_2^2-2(1-\alpha)\boldsymbol{x}^T\boldsymbol{\nu}]}q(\boldsymbol{\nu})d\boldsymbol{\nu} \tag{2f}$$

$$= \rho(\boldsymbol{x})\mathcal{H}(\alpha,\boldsymbol{x}), \tag{2g}$$

where Eq. (2a) is obtained by the independence of $\boldsymbol{X}$ and $\boldsymbol{\epsilon}$ and the law of changing of random variable, Eq. (2b) and Eq. (2d) are obtained by the definition of the Gaussian distribution, Eq. (2c) is obtained by decomposing the inner square, Eq. (2e) is obtained by substituting $\boldsymbol{\tau} = (1-\alpha)\boldsymbol{\nu}$, and Eq. (2f) is obtained by using the law of changing of random variable again.

Then, the KL-divergence between p_T and the convex combination ω_T can be written as

$$KL(p_T||\omega_T) = \int p_T(\boldsymbol{x})\log\frac{p_T(\boldsymbol{x})}{\omega_T(\boldsymbol{x})}d\boldsymbol{x} \tag{3a}$$

$$= \int p_T(\boldsymbol{x})[\log\frac{p_T(\boldsymbol{x})}{\rho(\boldsymbol{x})} - \log\mathcal{H}(\alpha,\boldsymbol{x})]d\boldsymbol{x} \tag{3b}$$

$$= KL(p_T||\rho) - \int p_T(\boldsymbol{x})\log\mathcal{H}(\alpha,\boldsymbol{x})d\boldsymbol{x} \tag{3c}$$

$$= KL(p_T||\rho) + \mathcal{F}(\alpha) \tag{3d}$$

where Eq. (3a) is the definition of the KL-divergence, Eq. (3b) is obtained from Eq. (2g).

Part 2. With assumption (i) to (xii), by implementing the Theorem 1 in [24], we have

$$KL(p||\omega) \leqslant \mathcal{J}_{SM} + KL(p_T||\rho) + \mathcal{F}(\alpha). \tag{4}$$

Because, $\mathcal{F}(1) = 0$ and $\mathcal{F}'(1) = \mathbb{E}_{\boldsymbol{X}\sim q}[X]\mathbb{E}_{\boldsymbol{X}\sim p_T}[X]$, then by Taylor expansion, we have

$$\mathcal{F}(\alpha) = (\alpha-1)\mathbb{E}_{\boldsymbol{X}\sim q}[\boldsymbol{X}]\mathbb{E}_{\boldsymbol{X}\sim p_T}[\boldsymbol{X}] + o((\alpha-1)^2). \tag{5}$$

Thus, with assumption (xiii), as α goes to 1, $\mathcal{F}(\alpha)$ converges to 0. □

References

[1] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. Invariance principle meets information bottleneck for out-of-distribution generalization. *arXiv:2106.06607*, 2021. 1

[2] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *CVPR*, pages 18187–18197, 2022. 3

[3] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. ILVR: conditioning method for denoising diffusion probabilistic models. In *ICCV*, 2021. 3

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[5] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 3

[6] Jin Gao, Jialing Zhang, Xihui Liu, Trevor Darrell, Evan Shelhamer, and Dequan Wang. Back to the source: Diffusion-driven test-time adaptation. *CoRR*, 2022. 3

[7] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021. 3

[8] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, Dawn Song, Jacob Steinhardt, and Justin Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *ICCV*, 2021. 1

[9] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. 4

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NIPS*, 2020. 3

[11] Daehee Kim, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. *arXiv:2104.09841*, 2021. 1

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3

[13] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017. 4

[14] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 3

[15] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017. 2

[16] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In *ICML*, 2022. 3

[17] Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele, and Bernhard Schölkopf. Learning explanations that are hard to vary. *arXiv:2009.00329*, 2020. 1

[18] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *CoRR*, 2022. 3

[19] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 3

[20] Yangjun Ruan, Yann Dubois, and Chris J. Maddison. Optimal representations for covariate shift. In *ICLR*, 2022. 1

[21] Shiori Sagawa, Pang Wei Koh, Tatsunori B. Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv:1911.08731*, 2019. 1

[22] Soroosh Shahtalebi, Jean-Christophe Gagnon-Audet, Touraj Laleh, Mojtaba Faramarzi, Kartik Ahuja, and Irina Rish. Sand-mask: An enhanced gradient masking strategy for the discovery of invariances in domain generalization. *arXiv:2106.02266*, 2021. 1

[23] Yuge Shi, Jeffrey Seely, Philip H. S. Torr, Siddharth Narayanaswamy, Awni Y. Hannun, Nicolas Usunier, and Gabriel Synnaeve. Gradient matching for domain generalization. In *ICLR*, 2022. 1

[24] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *NIPS*, 2021. 6, 7

[25] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *NeurIPS*, 2019. 1

[26] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv:2001.00677*, 2020. 1