Distribution Shift Inversion for Out-of-Distribution Prediction

Runpeng Yu Songhua Liu Xingyi Yang Xinchao Wang[†]
National University of Singapore
{r.yu,songhua.liu,xyang}@u.nus.edu xinchao@nus.edu.sg

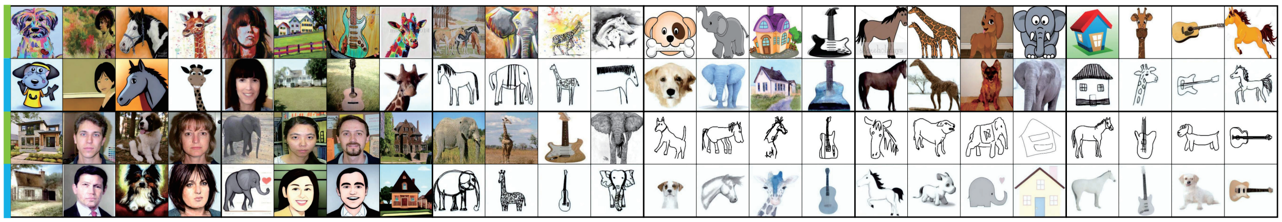


Figure 1. Transformed OoD samples from PACS dataset. Odd rows show the original OoD images, and even rows show their transformation results to the source distribution, obtained by the proposed DSI. Please zoom in for better visualization.

Abstract

Machine learning society has witnessed the emergence of a myriad of Out-of-Distribution (OoD) algorithms, which address the distribution shift between the training and the testing distribution by searching for a unified predictor or invariant feature representation. However, the task of directly mitigating the distribution shift in the unseen testing set is rarely investigated, due to the unavailability of the testing distribution during the training phase and thus the impossibility of training a distribution translator mapping between the training and testing distribution. In this paper, we explore how to bypass the requirement of testing distribution for distribution translator training and make the distribution translation useful for OoD prediction. We propose a portable Distribution Shift Inversion (DSI) algorithm, in which, before being fed into the prediction model, the OoD testing samples are first linearly combined with additional Gaussian noise and then transferred back towards the training distribution using a diffusion model trained only on the source distribution. Theoretical analysis reveals the feasibility of our method. Experimental results, on both multiple-domain generalization datasets and single-domain generalization datasets, show that our method provides a general performance gain when plugged into a wide range of commonly used OoD algorithms. Our code is available at https://github.com/yu-rp/Distribution-Shift-Iverson.

1. Introduction

The ubiquity of the distribution shift between the training and testing data in the real-world application of machine learning systems induces the study of Out-of-Distribution (OoD) generalization (or domain generalization). [18, 64, 71, 79] Within the scope of OoD generalization, machine learning algorithms are required to generalize from the seen training domain to the unseen testing domain without the independent and identically distributed assumption. The bulk of the OoD algorithms in previous literature focuses on promoting the generalization capability of the machine learning models themselves by utilizing the domain invariant feature [2, 13, 36], context-based data augmentation [47, 74], distributionally robust optimization [59], subnetwork searching [81], neural network calibration [70], etc.

In this work, orthogonal to enhancing the generalization capability of the model, we consider a novel pathway to OoD prediction. On the way, the testing(target) distribution is explicitly transformed towards the training(source) distribution to straightforwardly mitigate the distribution shift between the testing and the training distribution. Therein, the OoD prediction can be regarded as a two-step procedure, (1) transferring testing samples back towards training distribution, and (2) drawing prediction. The second step can be implemented by any OoD prediction algorithm. In this paper, we concentrate on the exploration of the first step, the distribution transformation.

Unlike previous works on distribution translation and domain transformation, in which certain target distribution is accessible during the training phase, here the target distribution is arbitrary and unavailable during the training. We term this new task as Unseen Distribution Transformation (UDT), in which a domain translator is trained on the source distribution and works to transform unseen target distribution towards the source distribution. The uniqueness, as well as the superiority, of UDT is listed as followings.

[†]Corresponding author.

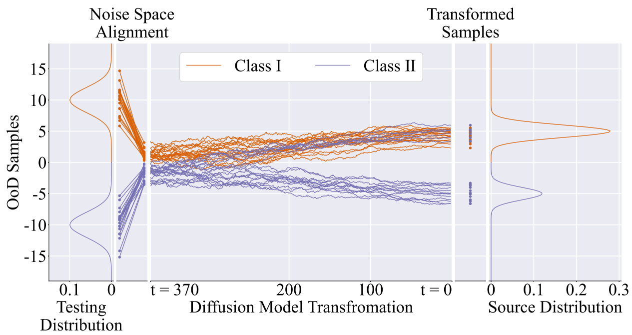


Figure 2. Using a Diffusion Model to solve the one-dimensional UDT. OoD samples are transformed to the source distribution with limited failure of label preservation.

- UDT puts no requirements on the data from both source and testing distribution like previous works do. This is practically valuable, because the real-world testing distributions are uncountable and dynamically changing.
- UDT is able to transform various distributions using only one model. However, the previous distribution translator works for the translation between certain source and target distributions. With a different source-target pair, a new translator is required.
- Considering the application of UDT in OoD prediction, it is free from the extra assumptions commonly used by the OoD generalization algorithms, such as the multi-training domain assumption and the various forms of the domain invariant assumption.

Despite the advantages, the unavailability of the testing distribution poses new difficulty. Releasing this constraint, the idea of distribution alignment is well established in domain adaptation (DA). Wherein, a distribution translator is trained with the (pixel, feature, and semantic level) cycle consistency loss [23, 38, 82]. However, the training of such distribution transfer modules necessitates the testing distribution, which is unsuitable under the setting of OoD prediction and makes the transplant of the methods in DA to OoD even impossible.

To circumvent the requirement of testing distribution during training time, we propose a novel method, named Distribution Shift Inversion (DSI). Instead of using a model transferring from testing distribution to training distribution, an unconditional generative model, trained only on the source distribution, is used, which transfers data from a reference noise distribution to the source distribution. The method operates in two successive parts. First, the OoD target distribution is transferred to the neighborhood of the noise distribution and aligned with the input of the generative model, thereafter we refer to this process as the forward transformation. The crux of this step is designing to what degree the target distribution is aligned to the noise distribution. In our implementation, the forward transformation is conducted by linearly combining the OoD samples and random noise with the weights controlling the alignment.Then, in the second step, the outcome of the first step is transferred towards the source distribution by a generative model, thereafter we refer to this process as the backward transformation. In this paper, the generative model is chosen to be the diffusion model [21]. The superiority of the diffusion model is that its input is the linear combination of the source sample and the noise with varying magnitude, which is in accord with our design of the forward transformation and naturally allows strength control. Comparatively, VAE [28] and GAN [16] have a fixed level of noise in their input, which makes the forward transformation strength control indirect. Our theoretical analyses of the diffusion model also show the feasibility of using the diffusion model for UDT.

Illustrative Example. A one-dimensional example is shown in Fig. 2.The example considers a binary classification problem, in which, given label, the conditional distributions of the samples are Gaussian in both the source and the testing domain. The testing distribution is constructed to be OoD and located in the region where the source distribution has a low density. The diffusion model is trained only on the source distribution. Passing through the noise space alignment and diffusion model transformation, the OoD samples are transformed to the source distribution with limited failure of label preservation.

Transformed Images. Fig. 1 shows some transformation results of OoD images towards the source distribution. The observation is twofold. (1) The distribution (here is the style) of the images is successfully transformed. All of the transferred images can be correctly classified by the ERM model trained on the source domain. (2) The transformed images are correlated to the original images. Some structural and color characteristics are mutually shared between them. This indicates that the diffusion model has extracted some low-level information and is capable to preserve it during the transformation. We would like to highlight again that, during the training, the diffusion model is isolated from the testing domain.

Our contributions are therefore summarized as:

- We put forward the unseen distribution transformation (UDT) and study its application to OoD prediction.
- We offer theoretical analyses of the feasibility of UDT.
- we propose DSI, a sample adaptive distribution transformation algorithm for efficient distribution adaptation and semantic information preservation.
- We perform extensive experiments to demonstrate that our method is suitable for various OoD algorithms to achieve performance gain on diversified OoD benchmarks. On average, adding in our method produces 2.26% accuracy gain on multi-training domain generalization datasets and 2.28% on single-training domain generalization datasets.

2. Related Work

2.1. Out-of-Distribution Generalization

To achieve OoD generalization, diverse methods have been proposed.

Causal inference methods extract the invariant feature among training domains and build up a unified predictor for all domains. For example, CNBB [19] down-weights the samples which introduce big changes to the feature space by evaluating the causal effect of each sample; ICP [51] find the subset of features satisfying the property that the conditional probability of the target given this set of features is invariant among the domains; IRM [2] regularizes the ERM loss with the norm of the gradient of loss corresponding to the latent features; IB-IRM [1] extends the IRM to the case, where the invariant features are only partially informative, by adding an extra entropy loss term of the latent feature.

Context-based data augmentation methods enrich the training feature space by linear interpolation between the features of inter-domain and intra-domain samples [74]; or combining the normalized feature of one sample with the mean and variance of the feature of another sample [47].

Distributional Robust Optimization method, like Group-DRO [59], theoretically formulates the OoD generalization as a min-max optimization and practically up-weights the domains with large losses in an online learning style to guarantee the model fit on all domains.

Feature alignment methods learn a unified feature representation among all training domains with the assumption that this representation is shared by the testing domain data. For example, DANN [13] and CDANN [36] align the marginal and conditional feature distribution by domain adversarial training, respectively; CORAL [67] matches the covariance of the features in every training domain; MASF [11] proposes model-agnostic episodic learning to regularize the semantic structure of the feature space; using the contrastive learning as the regularization term, SelfReg [27] minimizes the distance between the features of the samples within the sample class; CAD and ConCAD [58] minimize the mutual information between the feature and domain variable as well as the negative mutual information between the feature of original samples and the feature of augmented samples.

Context feature separation methods recognize the context feature by an extra context discriminator and disentangle them from the category feature by orthogonality penalty [4].

Gradient manipulation methods are designed based on the intuitions that (1) to train a unified predictor, only the gradient common across all domains should be used; or the intuition that (2) the spurious correlation, which leads to larger gradients, is easier to be fitted to and prevent the algorithm from learning other features. For example, IGA [30] and SD [52] penalize the variance of the gradients and the l_2 norm of the logits, respectively; RSC [25] only takes the samples whose gradients are small into consideration; MLDG [34] updates the parameters only when there are performance gains in two separated parts of the training dataset; ANDMask [50] and its ReLU-smoothed version SANDMask [63] only update the parameter whose gradients in most of the domains have the same sign.

Different from the previous works, DSI tackles the OoD generalization problem by test time sample adaption. Instead of increasing the OoD generalization ability of the model, DSI aims to mitigate the distribution shift of the testing samples by shifting them back towards the source distribution. Though input enhancing is investigated in dataset distillation [39] and domain adaptation [23, 38, 82], we are the first to test this idea in OoD generalization. Besides its novelty, DSI is orthogonal to the previous algorithms, which allows it to be portably used as a common technique for OoD tasks.

2.2. Diffusion Model

Neural network reuse has drawn recent interest [75–78]. In this work, we consider reusing pre-trained diffusion models. With promising performance and theoretical explanation, the diffusion model has been intensively investigated in the field of time series generalization/imputation/prediction [29, 32, 56, 68], image generalization/editing/inpainting [3, 6–8, 41, 44, 54, 61], text generalization/modeling [35, 80], text2image generalization [17, 48, 55, 60], text2speech generalization [53], video generalization [22], graph generalization [24, 73], 3D point cloud generalization [42, 43, 83]. Besides, these generation tasks, diffusion model conduces to the downstream tasks such as image segmenting [5], testing time sample adaptation for corrupted images [14], and adversarial attack defense [49]. In DSI, we explore a new application of the diffusion model in which it enhances the OoD prediction by transferring the OoD samples towards the source distribution.

2.3. Prediction Confidence Estimation

In DSI, the prediction confidence score is used for selecting poorly predicted samples and achieving an adaptive distribution shift inversion at the sample level. By satisfying the partial order relationship of uncertainty, commonly used loss functions, such as softmax cross entropy, are suitable confidence metrics. [31] Other softmax-based scores, such as the maximum class probability [15, 20], true class probability [10] and the KL-divergence between the softmax distribution and the uniform distribution [20] are also widely used. Based on the Bayesian method, Monte Carlo dropout [12], use the standard uncertainty criteria (e.g., variance, entropy) of the stochastic network predictions as the confidence score. Taking a modified nearest-neighbor classifier as the reference, trust score [26] uses a minimal distance ratio as the confidence score. We note that measurements in the field of OoD detection [37, 40, 46, 57, 62] and sample difficulty estimation may also serve as proper filtering metrics for DSI. But, in this paper, we prefer to establish a new OoD prediction framework and leave the discovery of the potential filtering metrics as future work.

3. Preliminary

The task of sampling an instance $\boldsymbol{x}$ of a random variable $\boldsymbol{X} \in \mathcal{R}^d$ with distribution $p(\boldsymbol{x})$ is generally intractable due to the fact that $p(\boldsymbol{x})$ is complex or unknown. Diffusion model, such as ScoreFlow [66] and DDPM [21], first samples from an simple distribution $\rho(\boldsymbol{x})$ and then iteratively transforms sample $\boldsymbol{x}$ to makes its distribution consistent with $p(\boldsymbol{x})$.

Under the continuous time setting, diffusion model formulates the forward transformation from $p(\boldsymbol{x})$ to $\rho(\boldsymbol{x})$ as a stochastic process $\{x_t\}_{t=0}^T$ following the Stochastic Differential Equation (SDE)

$$d\boldsymbol{x} = f(\boldsymbol{x}, t)dt + g(t)d\boldsymbol{w}, \tag{1}$$

where $f(\cdot, t) : \mathcal{R}^d \rightarrow \mathcal{R}^d$ is named drift coefficient, $g(t) \in \mathcal{R}$ is named diffusion coefficient, and $\boldsymbol{w}$ is the Wiener process whose derivative $d\boldsymbol{w}$ is characterized by a standard Gaussian random variable (white Gaussian noise). The corresponding marginal distribution $p_t(\boldsymbol{x})$ at time t satisfies $p_0(\boldsymbol{x}) = p(\boldsymbol{x})$ and $p_T(\boldsymbol{x}) \approx \rho(\boldsymbol{x})$. To sample from $p(\boldsymbol{x})$, the diffusion model first samples from $\rho(\boldsymbol{x})$ and then transforms $\boldsymbol{x}$ backward according to the inverse SDE associated with Eq. (1)

$$d\boldsymbol{x} = [f(\boldsymbol{x}, t) - g^2(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})]dt + g(t)d\bar{\boldsymbol{w}}, \tag{2}$$

where dt is the negative time flow, evolving from $t = T$ to $t = 0$, and $\bar{\boldsymbol{w}}$ is the Wiener process in the inverse time direction whose derivative $d\bar{\boldsymbol{w}}$ is again a standard Gaussian random variable because of the Gaussian increment characterization of Wiener process. In Eq. (2), $\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})$ is the unknown score function of $\boldsymbol{x}$ at time t, which is estimated by a neural network $s_{\boldsymbol{\theta}}(\boldsymbol{x}, t)$ with the weighted score matching loss

$$\begin{aligned} &\mathcal{J}_{SM}(\boldsymbol{\theta}; \lambda(\cdot)) \\ &\quad = \frac{1}{2}\int_0^T \mathbb{E}_{p_t(\boldsymbol{x})}[\lambda(t)||\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}) - s_{\boldsymbol{\theta}}(\boldsymbol{x}, t)||_2^2]dt. \end{aligned} \tag{3}$$

4. Unseen Distribution Transformation

First, we motivate our method by analyzing why the diffusion model helps promote the OoD generalization.

Here, we proof that, by feeding a linear combination of the OoD samples and the standard Gaussian noise to a diffusion model, the OoD samples can be transformed to the ID samples.

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

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

Theorem 1 proves that the testing distribution can be transformed to the source distribution and the convergence is controlled by α. The first terms in the inequality are introduced by the pretrained diffusion model. The loss term $\mathcal{J}_{SM}$ is small after sufficient training and can be reduced if the training procedure is further optimized. The KL-divergence term $KL(p_T||\rho)$ indicates the distance between the real distribution achieved by the forward transformation and the manually chosen standard Gaussian distribution, which is monotonically decreasing as the T goes larger and converges to 0 as the T is sufficient large. The third term (detailed in Appendix) is introduced by the distribution of the OOD testing samples, which is controlled by α and converges to 0 as α goes to 1.

4.1. Implementation

In this part, we describe DSI to realize the idea of shifting testing image towards the training distribution to boost the OoD generalization. The entire workflow is illustrated in Fig. 3.

In the previous analysis, by constructing a linear combination of the OoD input and the noise, the alignment between the testing and the training distributions is converted to an alignment problem in the noise space. To be consistent with the diffusion model literature, we rewrite the linear combination as $\hat{x} = \beta x + \alpha\boldsymbol{\epsilon}$. By adjusting the coefficients α, β and the starting time s, the distance between the distribution of $\hat{x}$ (i.e., $w(\hat{x})$) and the input distribution of the diffusion model (during the training time) at time s (i.e., $p_s(x)$) is controlled. To better match $w(\hat{x})$ to $p_s(x)$, utilizing the common practice [21, 65] that x_s is a designed to be a linear combination of x and Gaussian noise, we keep s as a hyperparameter and calculate α and β from s. Conditioned on the diffusion model, the maps from s to α and β vary. Taking DDPM as an example, the calculations can be written as $\alpha = \sqrt{1 - \prod_{l=1}^{s}(1 - \sigma_l)}$ and $\beta = \sqrt{\prod_{l=1}^{s}(1 - \sigma_l)}$, where σ_l is the standard deviation of the noise in the forward process at time l.

Next, we look into the schedule of starting time s. Two factors are taken into consideration. First is the time efficiency. The generation of the diffusion model requires an

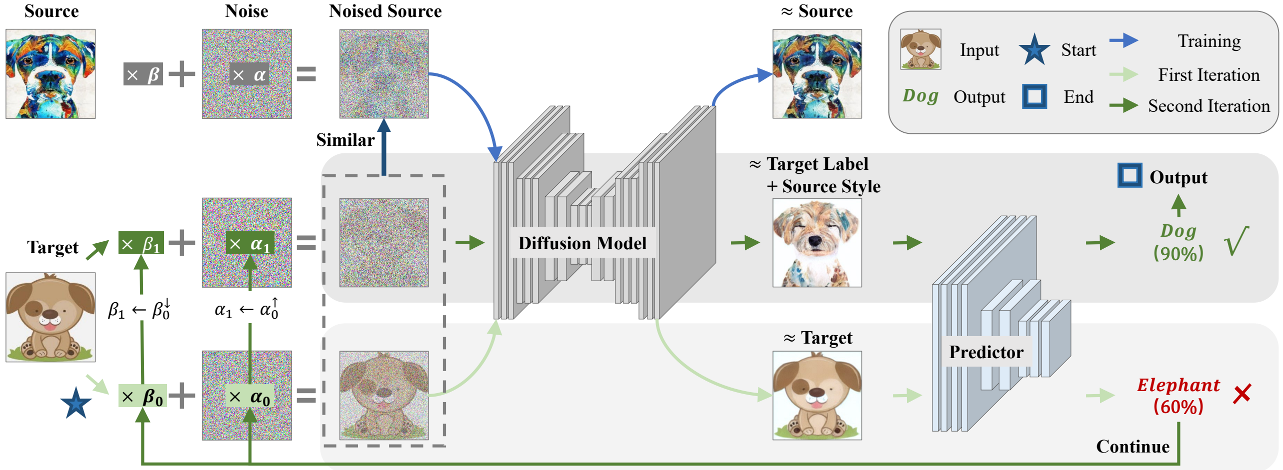


Figure 3. Illustrative example of DSI finished in two iterations. In the first iteration, insufficient transformation leads to wrong prediction with low confidence. Our algorithm rejects the prediction and continues the second iteration. In the second iteration, with proper transformation, the correct prediction is drawn and accepted. Then, the algorithm finished.

iterative sampling in which the total sampling steps (thus the sampling time) are controlled by the starting time s. Though providing more sufficient distribution transformation, using a large s is time-consuming and inappropriate for the online testing environment. [9, 72] Second is the sample differences. As discovered in image editing [45] using the diffusion model, the starting time controls the faithfulness and reality of the generated images. Analogically, in using the diffusion model to transfer the OoD data, s is a controller of the degree of distribution transformation and the preservation of semantic information, which are both crucial to OoD prediction. In practice, the distance from different samples to the training distribution is different and the difficulty of the semantic label preservation varies among the samples, which makes a uniform starting time schedule improper.

Thus, in DSI, a sample adaptive schedule of the starting time index is used, in which s increases gradually. Specifically, given a predefined starting time sequence $\{s_l\}_{l=0}^{L}$, where $s_0 < s_1 < \cdots < s_L$, for each OoD sample x, DSI begins with the smallest s_0, uses the diffusion model to transfer the combination $\hat{x}$, and lets the predictor output the prediction result h together with the confidence score c of the prediction. If the c is greater than a predefined threshold k, the prediction is accepted. Otherwise, DSI rejects the prediction, move to the next time index, and repeats the above procedure (transfer, prediction, and then using confidence score to make a judgment). If the prediction confidence score still does not meet the threshold at time index s_L, the last prediction is accepted. By doing so, DSI allocates a small starting time index to the samples close to the source distribution and a larger one to the samples far from the

Algorithm 1 Distribution Shift Inversion

Input: Predictor: f, Diffusion Models: $\{g_m\}_{m=1}^{M}$, Function for calculating α and β: ϕ, Ensemble function: ξ, Confidence score function: ψ
Input: Starting time series: $\{s_l\}_{l=0}^{L}$, Threshold: k
Input: OoD sample: x

$h_0 \leftarrow f(x)$ ▷ Draw prediction from original sample
for l **in** $1, \cdots, L$ **do**
  $\alpha, \beta \leftarrow \phi(t_l)$
  $\hat{x} \leftarrow \beta x + \alpha \epsilon$
  for m **in** $1, \cdots, M$ **do**
    $\tilde{x} \leftarrow g_m(\hat{x})$ ▷ Transfer
    $h_m \leftarrow f(\tilde{x})$
  end for
  $h \leftarrow \xi(h_0, \cdots, h_M)$ ▷ Ensemble
  if $\psi(h) > k$ **then** ▷ Confidence Filtering
    break
  end if
end for
output prediction h ▷ Accept the last prediction

source distribution. Thus, this adaptation procedure avoids the use of a uniformly small time step which causes some samples to not be fully transferred, or a uniformly large time steps to cause the diffusion model to take too long to proceed. In our experiments, we use the maximum class probability as the confidence score and set the confidence threshold as hyperparameter.

Usually, under the setting of OoD prediction, multiple training domains are available. The majority of the OoD

Dataset	PACS					OfficeHome				
Testing Domain	A	C	P	S	Average	A	C	P	R	Average
ERM	76.24±0.44	67.90±0.64	94.47±0.44	60.32±0.85	74.73±0.59	60.81±0.51	54.17±0.17	74.80±0.35	76.11±0.40	66.47±0.36
ERM*	**78.52±0.81**	**70.90±0.76**	**95.80±0.16**	**67.97±0.68**	**78.30±0.60**	61.15±0.56	**55.96±0.76**	**75.55±0.23**	76.56±0.29	**67.31±0.46**
ANDMask [50]	67.35±0.51	63.22±2.23	93.98±0.26	60.35±1.05	71.23±1.01	60.84±0.60	55.73±0.60	73.80±0.44	75.88±0.08	66.56±0.43
ANDMask*	**69.43±0.40**	**68.23±2.29**	**94.99±0.24**	**69.63±0.57**	**75.57±0.88**	61.18±0.65	**56.80±0.41**	74.35±0.40	**76.40±0.54**	67.18±0.50
CAD [58]	75.42±0.51	68.20±0.64	95.02±0.55	59.77±2.14	74.60±0.96	60.71±0.64	53.39±0.18	74.41±0.29	75.36±0.39	65.97±0.38
CAD*	76.11±0.81	**71.81±0.59**	**96.09±0.16**	**69.56±1.69**	**78.39±0.81**	61.04±0.63	**55.08±0.44**	**75.20±0.29**	**76.14±0.28**	**66.87±0.41**
CondCAD [58]	75.55±0.38	67.77±0.50	95.57±0.12	59.99±1.90	74.72±0.73	60.74±0.37	55.27±0.83	74.64±0.26	75.42±0.20	66.52±0.42
CondCAD*	**77.12±0.36**	**71.61±0.37**	**96.03±0.09**	**69.79±1.96**	**78.64±0.70**	61.15±0.28	56.09±0.58	**75.52±0.41**	**76.43±0.17**	**67.30±0.36**
GroupDRO [59]	69.56±1.01	63.77±3.74	93.16±0.77	64.16±1.78	72.66±1.83	60.32±0.26	53.12±0.64	71.61±0.66	75.68±0.21	65.18±0.44
GroupDRO*	**71.61±0.81**	67.45±2.75	93.78±0.40	**71.88±1.02**	**76.18±1.25**	60.73±0.26	**55.83±0.79**	**73.01±0.92**	**76.14±0.18**	**66.43±0.54**
IB_ERM [1]	77.18±0.20	69.76±0.24	94.82±0.80	61.26±0.58	75.76±0.46	58.30±0.78	54.39±0.10	70.36±0.05	72.02±0.44	63.77±0.34
IB_ERM*	**79.07±0.20**	**71.88±0.44**	95.83±0.30	**69.95±0.88**	**79.18±0.46**	58.66±0.81	**55.66±0.39**	**71.29±0.01**	**72.90±0.34**	**64.63±0.39**
IB_IRM [1]	75.26±1.09	67.77±0.97	94.76±0.45	60.25±2.40	74.51±1.23	58.01±0.44	52.60±1.00	72.75±0.60	74.90±0.65	64.57±0.67
IB_IRM*	77.31±1.08	**70.70±0.63**	**95.96±0.12**	**68.03±1.67**	**78.00±0.88**	58.37±0.39	53.68±0.44	72.95±0.84	75.55±0.79	65.14±0.62
Mixup [74]	71.29±0.49	66.11±2.32	94.08±0.47	64.81±2.88	74.07±1.54	61.46±0.20	55.83±0.65	74.06±0.44	76.73±0.41	67.02±0.43
Mixup*	**73.18±0.17**	69.63±2.35	94.56±0.40	**71.45±2.74**	**77.21±1.42**	**62.25±0.31**	**57.85±0.85**	**75.10±0.40**	77.21±0.26	**68.10±0.46**
SANDMask [63]	67.35±0.51	62.50±3.85	94.17±0.62	64.58±0.58	72.15±1.39	61.78±0.36	55.24±0.12	73.34±0.40	75.68±0.37	66.51±0.31
SANDMask*	**69.43±0.40**	67.42±2.96	94.73±0.42	**72.66±0.63**	**76.06±1.10**	62.14±0.35	**56.58±0.28**	73.50±0.23	76.27±0.29	**67.12±0.29**
SelfReg [27]	76.86±0.08	68.62±0.74	96.09±0.14	61.39±0.81	75.74±0.44	62.50±0.76	56.09±0.23	75.52±0.38	75.75±0.24	67.47±0.40
SelfReg*	**78.61±0.41**	**71.55±0.74**	96.39±0.21	**71.19±0.21**	**79.44±0.39**	62.90±0.82	**57.52±0.14**	**76.37±0.37**	**76.79±0.20**	**68.40±0.38**
Average Gain	1.83±0.43	3.55±0.85	0.80±0.34	8.52±1.06	3.68±0.67	0.41±0.13	1.52±0.52	0.75±0.35	0.69±0.21	0.84±0.30

Table 1. The average accuracy $\pm$ the standard deviation of base algorithms w/o our method on PACS and OfficeHome datasets. The performance is generally boosted when our method is plugged in, whichever base algorithm is used.

prediction algorithms are based on this multiple-training domain assumption to extract stable features and establish a uniform predictor. Consistent with this common setting, for the multiple training domain problem, we use a unified predictor f trained on all available training domains. However, a mixture of multiple distributions increases the difficulty of training for the diffusion model, which leads to a slower convergence and domain collapse (only generating samples from easy domains). Given consideration to these undesired phenomena, for the multiple training domain problem, we train individual diffusion model g_m, where $m = 1, \cdots, M$ is the training domain index, on each training domain, use every g_m to transfer the OoD sample, and then obtain the prediction h_m from every transferred sample using f. Finally, we ensemble the predictions $\{h_m\}_{m=1}^M$ together to get the overall prediction. In our experiments, the ensemble is conducted by averaging the logits (the inputs of the softmax layer) in the predictor neural network.

The overall framework of DSI is shown in Algorithm 1.

5. Experiments

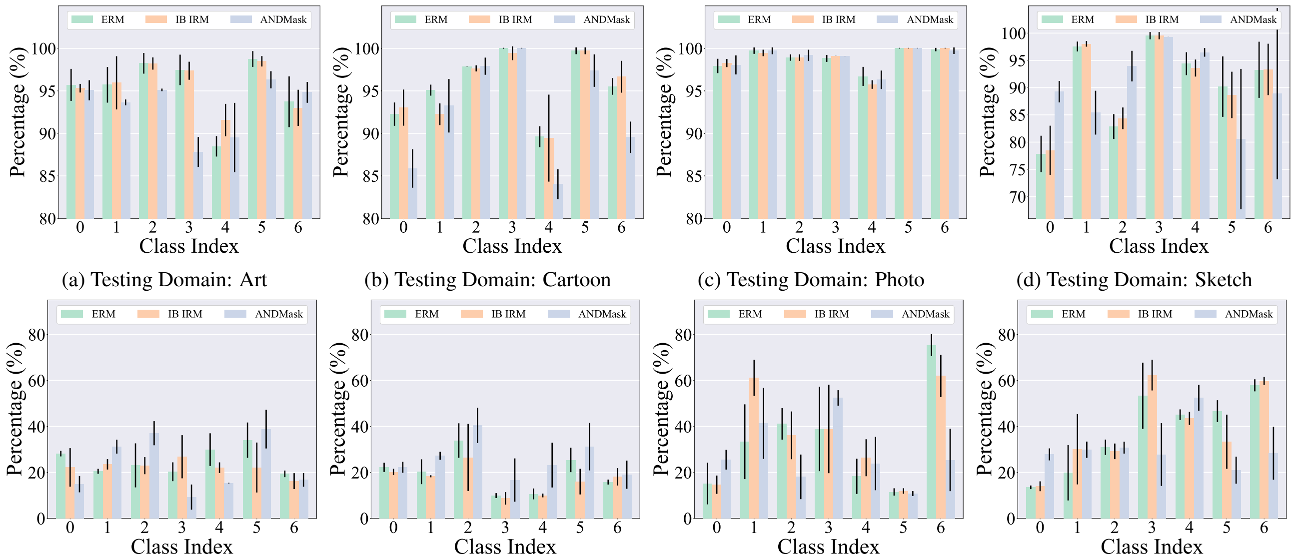
Datasets. In the main text, experiments on the following datasets are reported. More experiments can be found in supplementary materials. (1) PACS [33] contains 999,1 colored images from 7 classes and 4 domains (art painting, cartoon, photo, and sketch); (2) Office-Home [69] contains around 15,500 images from 65 different classes and 4 domains (artistic, clip art, product, and real-world).

Base Methods. Based on the taxonomy in Sec. 2, we select one (or two) up-to-date or representative algorithm(s) from each category as the base methods in the experiments with multiple training domains. For experiments with a single training domain, we adjust the list of base methods by only choosing the algorithms without multi-domain assumption. For each experimental configuration (dataset and training-testing split), the hyperparameters of each algorithm are optimized based on the performance metric on the testing set over 10 random searches. The searching regions are reported in supplementary materials. Fixing the optimal hyperparameters, we repeat the experimental pipeline three times and report the average results together with the standard deviations to alleviate the influence of the lucky weight initialization and dataset split.

In Tabs. 1 and 2, 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.

5.1. Multi-training Domain Generalization

The experiments for multi-training domain generalization are conducted on PACS and OfficeHome. For each dataset, one domain is left as the testing domain and the other three are used for training. The performance comparison of base methods with and without our method are listed in Tab. 1. Our method generally enhances the OoD prediction of all types of base methods by 3.68% on PACS and 0.84% on OfficeHome, which indicates DSI successfully closes the distance between the OoD samples and the training distribution. The significance of the improvements varies among different leave-one-out settings, because the accuracy of the base methods varies. When the accuracy of the base method is low (e.g., the average accuracy of the base methods is 61.88% for S of PACS), more (8.22%) improvement is achieved. When



(a) Testing Domain: Art (b) Testing Domain: Cartoon (c) Testing Domain: Photo (d) Testing Domain: Sketch

(e) Testing Domain: Art (f) Testing Domain: Cartoon (g) Testing Domain: Photo (h) Testing Domain: Sketch

Figure 4. The percentage of correct prediction when using our method among the samples which are correctly predicted by the base method (in Fig. 4a to Fig. 4d), or wrongly predicted by the base method (in Fig. 4e to Fig. 4h). The plot is separately drawn for each leave-one-out setting and each class.

the accuracy of the base method is high (e.g., the average accuracy of the base methods is 94.61% for P of PACS), the absolute value of the improvement is less (0.89%). We also notice that the rise in performance on OfficeHome dataset is less than the PACS, which can be attributed to that the larger number of classes in OfficeHome increases the hardness of the semantic label preservation of our method.

We further analyze whether the performance gain depends on a certain class or domain. We count, for each class in each testing domain, (1) the number of samples correctly predicted by both the base methods and the base methods with DSI($\#\ Both\ Correct$), (2) the number of samples correctly predicted by the base methods with DSI but wrongly predicted by base methods ($\#\ Only\ Ours\ Correct$), (3) the number of samples correctly predicted by the base methods ($\#\ Base\ Correct$), and (4) the number of samples wrongly predicted by the base methods ($\#\ Base\ Wrong$). Fig. 4 shows the preservation ratio calculated by $\frac{\#\ Both\ Correct}{\#\ Base\ Correct}$ and the correction ratio calculated by $\frac{\#\ Only\ Ours\ Correct}{\#\ Base\ Wrong}$. The preservation ratio indicates the percentage of the correct predictions that are still correct when our method is used. The correction ratio indicates the percentage of the wrong predictions that are corrected when our method is used. Averaging over testing domains and classes, 94.52% of the correct predictions are still correct when our method is used. The highest preservation ratio is 95.95% appearing when photo is the testing domain. Among the testing domains and the classes, our method can at least correct 8.57% wrong predictions and averagely 28.17%. These results prove the general effectiveness of our methods across different domains and classes.

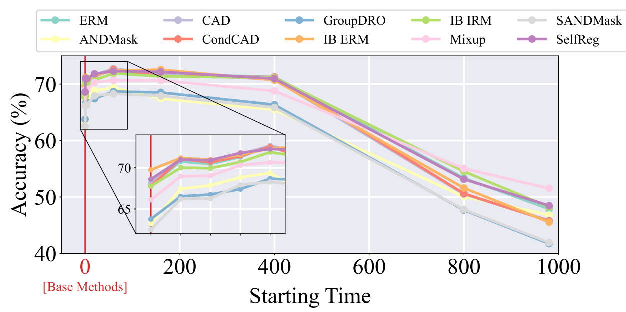


Figure 5. Accuracy v.s. Starting Time s on PACS dataset with the testing domain of "Cartoon".

5.2. Single-training Domain Generalization

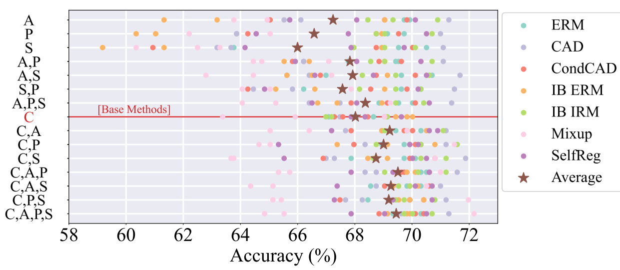
To cover the harder single training domain generalization, we do experiments on PACS datasets in which we use each domain as the training domain and test on the other three. The performance comparison of base methods with and without our method on the dataset with single training domains are listed in Tab. 2. With our method, irrelevant to the training and testing domain, general performance gains are achieved and the average performance gain is 2.42%.

5.3. Discussion on the starting time s

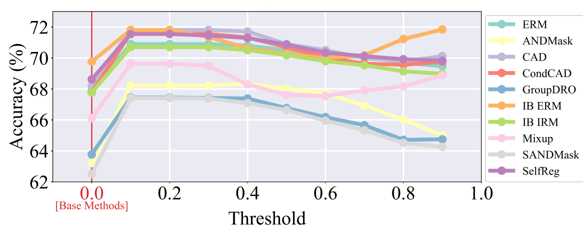
Transforming the distribution closer to the source distribution increases the confidence of the prediction based on the transformed image. On the other hand, the well-preserved label information guarantees the prediction of the generated image has the same label as its origin. However, increasing s to get further distribution transformation will destroy the label information, contrarily, decreasing s to preserve the

Dataset	PACS											
Training Domain	A			C			P			S		
Testing Domain	C	P	S	A	P	S	A	C	S	A	C	P
ERM	55.37±0.32	97.92±0.05	42.84±1.68	55.63±0.26	83.43±0.32	57.42±2.27	64.84±1.02	28.26±3.86	23.24±4.88	15.82±1.33	14.26±0.70	12.66±0.49
ERM*	**57.58±0.26**	97.90±0.03	**48.54±1.12**	**57.03±0.44**	83.98±0.56	**63.74±2.00**	65.92±0.37	32.75±3.78	25.68±4.78	16.41±1.51	**16.80±0.76**	12.92±0.20
CORAL	56.35±2.63	96.68±0.63	42.32±2.60	55.47±0.52	84.15±0.70	57.81±1.65	62.04±0.74	32.16±5.84	26.33±8.46	15.27±0.68	15.76±1.16	11.65±0.81
CORAL*	58.76±2.96	96.84±0.53	**47.56±1.16**	**56.87±0.09**	84.38±0.83	**64.65±2.01**	**63.90±1.01**	35.97±4.80	29.23±7.99	15.98±0.66	16.50±1.04	12.27±0.92
RSC	55.21±0.28	98.18±0.17	44.66±1.69	54.36±1.06	83.79±0.62	59.41±2.33	63.05±0.96	31.61±6.75	24.90±5.12	16.57±0.59	16.37±0.26	13.77±1.92
RSC*	**57.39±0.45**	98.24±0.15	**50.39±1.25**	55.70±0.74	83.95±0.56	**65.98±1.60**	64.13±1.64	35.29±5.77	28.22±4.87	17.68±0.68	**17.45±0.69**	14.49±1.88
SagNet	56.41±0.47	97.95±0.08	46.81±3.33	58.82±1.18	86.95±0.79	57.91±2.62	67.32±0.99	41.76±2.35	35.77±2.48	15.76±0.91	14.23±2.45	16.76±1.51
SagNet*	**58.85±0.51**	97.96±0.08	**53.48±3.30**	59.51±1.40	87.14±0.97	**64.71±3.50**	67.74±0.75	44.34±2.00	38.64±1.84	**18.03±0.67**	15.95±2.17	18.26±1.90
Average Gain	2.31±0.12	0.05±0.07	5.83±0.52	1.20±0.30	0.28±0.16	6.63±0.21	1.11±0.51	3.64±0.69	2.88±0.31	1.17±0.67	1.52±0.68	0.77±0.45

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



(a) Accuracy v.s. Ensemble Set



(b) Accuracy v.s. Threshold

Figure 6. Discussion about the influence of hyperparameters: Ensemble set (Fig. 6a) and Threshold (Fig. 6b) on the PACS dataset with the testing domain of "Cartoon". The y-axis label of Fig. 6a indicates the components of the ensemble set with the corresponding accuracy's show on the right. The row of "C" corresponds to the setting using the original testing images alone. The areas below(above) correspond to the ensembles with(without) the original testing images.

label information will limit the distribution transformation. Thus, an optimal s exists for each algorithm. (see Fig. 5)

5.4. Discussions on hyperparameters

Ensemble. We analyze whether only transforming the testing domain data to a subset of the training domains influences the performance gain or not. Fig. 6a shows the accuracy with different ensemble sets on PACS with the multi-training domain setting and Cartoon is the testing domain. Each row corresponds to one kind of ensemble. For example, row "C" shows the accuracy when the prediction is only based on the original testing image, which is the accuracy of the base methods; row "C,A" shows the accuracy when the prediction based on the original testing image and prediction based on the image transformed towards the domain Art are considered together. As shown from the figure, including more components in the ensemble set increases the average performance and reduces its variance; ensembles without the original prediction can perform comparably with the base method; the ensembles with the original prediction and any (one, two, or all) of the predictions based on the transformed image is statistically better than the base methods.

Confidence Threshold. We analyze the influence of the confidence threshold k, and the performance with varying k are shown in 6b. Base methods are special cases when $k = 0$, their performances are shown along the red vertical line Though tuning k enlarges the performance gain, when $k > 0$, our method is activated and there are always performance improvements.

6. Conclusion

In this paper, we investigate a novel task, termed unseen distribution transformation, which aims at transforming the unseen distribution towards the seen distribution. We frame the first solution to unseen distribution transformation, in which the unseen distribution is first linearly combined with the Gaussian noise and then transformed by a diffusion model trained on the seen domain. By solving this task, we provide a new perspective for addressing the OoD prediction task by first closing the distance between the testing domain and the training domain and then drawing prediction. We propose the portable DSI, which conducts sample adaptive unseen distribution transformation to enhance the OoD prediction algorithms. Experimental results show that our method results in general performance gains when inserted into various types of base methods under multi-training and single-training domain generalization problems.

7. Acknowledgment

This project is supported by the Singapore Ministry of Education Academic Research Fund Tier 1 (Grantor's Reference Number: 23-0306-A0001), a project titled "Towards Robust Single Domain Generalization in Deep Learning".

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. 3, 6

[2] Martín Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv:1907.02893*, 2019. 1, 3

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

[4] Haoyue Bai, Rui Sun, Lanqing Hong, Fengwei Zhou, Nanyang Ye, Han-Jia Ye, S.-H. Gary Chan, and Zhenguo Li. Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. In *AAAI*, 2021. 3

[5] Dmitry Baranchuk, Andrey Voynov, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *ICLR*, 2022. 3

[6] Yaniv Benny and Lior Wolf. Dynamic dual-output diffusion models. In *CVPR*, 2022. 3

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

[8] Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models. In *CVPR*, 2022. 3

[9] Hyungjin Chung, Byeongsu Sim, and Jong Chul Ye. Come-closer-diffuse-faster: Accelerating conditional diffusion models for inverse problems through stochastic contraction. In *CVPR*, 2022. 5

[10] Charles Corbière, Nicolas Thome, Avner Bar-Hen, Matthieu Cord, and Patrick Pérez. Addressing failure prediction by learning model confidence. In *NeurIPS*, 2019. 3

[11] Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. *arXiv:1910.13580*, 2019. 3

[12] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016. 3

[13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 1, 3

[14] 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

[15] Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *NeurIPS*, 2017. 3

[16] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 2

[17] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *CVPR*, 2022. 3

[18] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *ICLR*, 2021. 1

[19] Yue He, Zheyan Shen, and Peng Cui. Towards non-i.i.d. image classification: A dataset and baselines. *Pattern Recognit.*, 2021. 3

[20] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. 3

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NIPS*, 2020. 2, 4

[22] Jonathan Ho, Tim Salimans, Alexey A. Gritsenko, William Chan, Mohammad Norouzi, and David J. Fleet. Video diffusion models. *CoRR*, 2022. 3

[23] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 2, 3

[24] Emiel Hoogeboom, Victor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3d. In *ICML*, 2022. 3

[25] Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *AAAI*, 2020. 3

[26] Heinrich Jiang, Been Kim, Melody Y. Guan, and Maya R. Gupta. To trust or not to trust A classifier. In *NeurIPS*, 2018. 3

[27] Daehee Kim, Seunghyun Park, Jinkyu Kim, and Jaekoo Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. *arXiv:2104.09841*, 2021. 3, 6

[28] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 2

[29] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *ICLR*, 2021. 3

[30] Masanori Koyama and Shoichiro Yamaguchi. When is invariance useful in an out-of-distribution generalization problem? *arXiv:2008.01883*, 2021. 3

[31] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, 2017. 3

[32] Max W. Y. Lam, Jun Wang, Dan Su, and Dong Yu. BDDM: bilateral denoising diffusion models for fast and high-quality speech synthesis. In *ICLR*, 2022. 3

[33] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. 6

[34] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. Learning to generalize: Meta-learning for domain generalization. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *AAAI*, 2018. 3

[35] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B. Hashimoto. Diffusion-lm improves controllable text generation. *CoRR*, 2022. 3

[36] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*, 2018. 1, 3

[37] Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018. 3

[38] Ming-Yu Liu, Thomas M. Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NeurIPS*, 2017. 2, 3

[39] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. *NeurIPS*, 2022. 3

[40] Weitang Liu, Xiaoyun Wang, John D. Owens, and Yixuan Li. Energy-based out-of-distribution detection. In *NIPS*, 2020. 3

[41] Andreas Lugmayr, Martin Danelljan, Andrés Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022. 3

[42] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *CVPR*, 2021. 3

[43] Zhaoyang Lyu, Zhifeng Kong, Xudong Xu, Liang Pan, and Dahua Lin. A conditional point diffusion-refinement paradigm for 3d point cloud completion. In *ICLR*, 2022. 3

[44] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 3

[45] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *ICLR*, 2022. 5

[46] Eric T. Nalisnick, Akihiro Matsukawa, Yee Whye Teh, and Balaji Lakshminarayanan. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *CoRR*, 2019. 3

[47] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *CVPR*, 2021. 1, 3

[48] 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

[49] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Animashree Anandkumar. Diffusion models for adversarial purification. In *ICML*, 2022. 3

[50] Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele, and Bernhard Schölkopf. Learning explanations that are hard to vary. *arXiv:2009.00329*, 2020. 3, 6

[51] Jonas Peters, Peter Bühlmann, and Nicolai Meinshausen. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):947–1012, 2016. 3

[52] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron C. Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *arXiv:2011.09468*, 2020. 3

[53] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail A. Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *ICML*, 2021. 3

[54] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *CVPR*, 2022. 3

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

[56] Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *ICML*, 2021. 3

[57] Jie Ren, Peter J. Liu, Emily Fertig, Jasper Snoek, Ryan Poplin, Mark A. DePristo, Joshua V. Dillon, and Balaji Lakshminarayanan. Likelihood ratios for out-of-distribution detection. In *NeurIPS*, 2019. 3

[58] Yangjun Ruan, Yann Dubois, and Chris J. Maddison. Optimal representations for covariate shift. In *ICLR*, 2022. 3, 6

[59] 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, 3, 6

[60] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding. *CoRR*, 2022. 3

[61] Vikash Sehwag, Caner Hazirbas, Albert Gordo, Firat Ozgenel, and Cristian Canton-Ferrer. Generating high fidelity data from low-density regions using diffusion models. In *CVPR*, 2022. 3

[62] Joan Serrà, David Álvarez, Vicenç Gómez, Olga Slizovskaia, José F. Núñez, and Jordi Luque. Input complexity and out-of-distribution detection with likelihood-based generative models. In *ICLR*, 2020. 3

[63] 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. 3, 6

[64] Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization: A survey. *CoRR*, 2021. 1

[65] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 4

[66] Yang Song, Conor Durkan, Iain Murray, and Stefano Ermon. Maximum likelihood training of score-based diffusion models. In *NIPS*, 2021. 4

[67] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In Dale Schuurmans and Michael P. Wellman, editors, *AAAI*, 2016. 3

[68] Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. CSDI: conditional score-based diffusion models for probabilistic time series imputation. In *NeurIPS*, 2021. 3

[69] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 6

[70] Yoav Wald, Amir Feder, Daniel Greenfeld, and Uri Shalit. On calibration and out-of-domain generalization. In *NeurIPS*, 2021. 1

[71] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. In *IJCAI*, 2021. 1

[72] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. In *ICLR*, 2022. 5

[73] Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. In *ICLR*, 2022. 3

[74] Shen Yan, Huan Song, Nanxiang Li, Lincan Zou, and Liu Ren. Improve unsupervised domain adaptation with mixup training. *arXiv:2001.00677*, 2020. 1, 3, 6

[75] Xingyi Yang, Jingwen Ye, and Xinchao Wang. Factorizing knowledge in neural networks. In *ECCV*, 2022. 3

[76] Xingyi Yang, Daquan Zhou, Jiashi Feng, and Xinchao Wang. Diffusion probabilistic model made slim. *CVPR*, 2023. 3

[77] Xingyi Yang, Daquan Zhou, Songhua Liu, Jingwen Ye, and Xinchao Wang. Deep model reassembly. *NeurIPS*, 2022. 3

[78] Jingwen Ye, Songhua Liu, and Xinchao Wang. Partial network cloning. *CVPR*, 2023. 3

[79] Nanyang Ye, Kaican Li, Lanqing Hong, Haoyue Bai, Yiting Chen, Fengwei Zhou, and Zhenguo Li. Ood-bench: Benchmarking and understanding out-of-distribution generalization datasets and algorithms. *arXiv:2106.03721*, 2021. 1

[80] Peiyu Yu, Sirui Xie, Xiaojian Ma, Baoxiong Jia, Bo Pang, Ruiqi Gao, Yixin Zhu, Song-Chun Zhu, and Ying Nian Wu. Latent diffusion energy-based model for interpretable text modelling. In *ICML*, 2022. 3

[81] Dinghuai Zhang, Kartik Ahuja, Yilun Xu, Yisen Wang, and Aaron C. Courville. Can subnetwork structure be the key to out-of-distribution generalization? In *ICML*, 2021. 1

[82] Sicheng Zhao, Bo Li, Xiangyu Yue, Yang Gu, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source domain adaptation for semantic segmentation. In *NeurIPS*, 2019. 2, 3

[83] Linqi Zhou, Yilun Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *ICCV*, 2021. 3