DomainFusion: Generalizing To Unseen Domains with Latent Diffusion Models

Yuyang Huang[1], Yabo Chen[1], Yuchen Liu[1], Xiaopeng Zhang[2(✉)], Wenrui Dai[1(✉)], Hongkai Xiong[1], and Qi Tian[2]

[1] Shanghai Jiao Tong University, Shanghai, China
[2] Huawei Inc., Shenzhen, China
{huangyuyang, chenyabo, liuyuchen6666, daiwenrui, xionghongkai}@sjtu.edu.cn
zxphistory@gmail.com, tian.qi1@huawei.com

1 More Discussion about Latent Distillation

1.1 A Recap of Differentiable Image Parameterizations

Images are conventionally parameterized as matrices of pixel values, with each pixel defined by its RGB intensities. However, in specific application contexts, this parametrization may not be the most efficient. Indeed, it is feasible to opt for a new set of parameters and establish a mapping relationship between these parameters and the image. Subsequently, we can compute an objective function leveraging the image, and if this mapping is differentiable, the objective function can facilitate efficacious updates to our selected novel parameters. The framework above is referred to as Differentiable Image Parameterizations [9], as is shown in Fig. 1.

1.2 Understanding Latent Distillation from Differentiable Image Parameterizations Perspective

Similarly, Latent Distillation can also be considered a novel form of differentiable image parameterizations, where we map the image $\mathbf{x}$ to a DG discriminative model's parameters θ. We exploit the perception capability of the DG model P_θ to corrupt the image $\mathbf{x}$ and subsequently extract the loss $\mathcal{L}_{\text{LD}}$ through a LDM for backpropagation, thereby updating the input parameters θ. Our method process shares similarities with other applications of differentiable image parameterizations, as it involves manipulating the attributes of the image to obtain high-quality results. However, in our method, the manipulation of image attributes is not the ultimate goal. Instead, it necessitates continuous improvement of the DG model's perception capability. By continuously propagating meaningful gradient information to minimize the objective function given by LDM, we can ultimately achieve a highly generalizable DG model.

Yuyang Huang and Yabo Chen—Equal contribution.
Correspondence to Xiaopeng Zhang and Wenrui Dai.

Algorithm 1: DomainFusion

Input: A latent diffusion model ϕ, the initialized DG model θ, the number of epochs N_e, the generation interval Itv, the number of candidates N_c, the weight in sampling strategy λ_s, the weight of loss λ_1, λ_2, λ_3.

Output: The trained DG model θ.

Initialize generate-dataset $\leftarrow$ raw-dataset;
for $i \leftarrow 1$ **to** N_e **do**
  $loss \leftarrow 0$; $l_{LD} \leftarrow 0$;
  if $i\%Itv == 0$ **then**
    foreach *generate-minibatch* **do**
      foreach $(\mathbf{x}, y)$ *in generate-minibatch* **do**
        $candidates = ldm(\mathbf{x}, y, N_c)$;
        decompose $\mathbf{x}$ into style (μ_0, σ_0) and content c_0;
        decompose *candidates* into style (μ, σ) and content c;
        $c^* \leftarrow \max_j \lambda_s cos(feature\ c_0, feature\ c^*) + (1 - \lambda_s) confidence(c^*, y, \theta)$;
        $\mu^*, \sigma^* \leftarrow \max_j KL(\mu_0, \mu_j) + KL(\sigma_0, \sigma_j)$;
        $\mathbf{x}_{gen} \leftarrow AdaIN(\mu^*, \sigma^*, c^*)$;
        $\mathbf{x} \leftarrow \mathbf{x}_{gen}$;
        $l_{LD} \leftarrow l_{LD} + \lambda_2 LD(\mathbf{x}, y, \theta, \phi)$;
  $loss \leftarrow loss + \lambda_1 l_{raw}$;
  $loss \leftarrow loss + \lambda_3 l_{gen}$;
  update θ;
return θ

Algorithm 2: LD

Input: An input image $\mathbf{x}$, the label y, the DG model θ, a latent diffusion model ϕ.

Output: the LD loss.

$\mathbf{x} \leftarrow confidence(\mathbf{x}, y, \theta) \cdot \mathbf{x}$; $l_{LD} \leftarrow 0$;
for $i \leftarrow 1$ **to** T **do**
  Sample t;
  Sample ϵ;
  $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x} + \sqrt{1 - \bar{\alpha}_t}\epsilon$;
  $l_{LD} \leftarrow l_{LD} + w[t] \left\| \epsilon - \epsilon_\theta \left(\mathbf{x}_t, \mathbf{c}_k \right) \right\|^2$
return l_{LD}

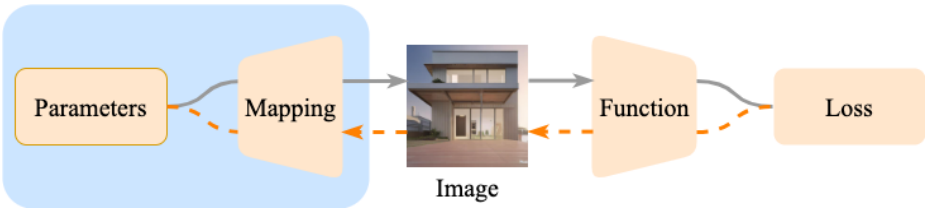


Fig. 1: The framework of Differentiable Image Parameterizations.

2 Pseudocode of DomainFusion

To present the algorithmic details of DomainFusion more clearly, we provide the following pseudocode in Alg 1 and Alg 2. Specifically, Alg 1 presents the overall architecture of DomainFusion and describes how DomainFusion trains the DG model by extracting loss from three aspects: the loss from the raw dataset, the loss from Latent Distillation, and the loss from online lightweight augmentation. Additionally, Alg 2 outlines the specific process of Latent Distillation, which involves extracting the Latent Distillation (LD) loss from the LDM using the given image and DG model.

3 Computational Costs Analysis

We provide computational costs analysis of our methods and other diffusion-based DG methods DSI [15] and CDGA [11] in Tab. 1. For DSI [15] and CDGA [11], we followed the experimental settings of their respective papers. Both DSI [15] and CDGA [11] use stable diffusion v1-4 according to their corresponding papers, and our method also employs the same LDM. Experiments are conducted on the Clipart domain of the DomainNet dataset. All computational costs were measured in terms of NVIDIA Tesla V100 GPU days.

For the finetuning time, we evaluated the time overhead based on the open-source DSI code [15]. DSI [15] requires finetuning a separate LDM on each of the five source domains. The total finetuning dataset size achieves 538,446 images. This process consumed approximately 22.44 GPU days, indicating a significant computational cost.

For the generation time, it can be determined based on the parameters of the generation process and the number of generated samples, as all three methods employ the same LDM. Regarding the generated sample quantity, DSI [15] requires generating over 244,000 images, CDGA [11] requires generating over 4 million images, while our method only needs to generate 120,000 images, significantly fewer than the former two methods. The reasons that we generate much smaller images than other methods are as follows. Firstly, we adopt an online generation strategy, deviating from the conventional offline generation approaches employed in other methods. Besides, in our framework the generation process is executed every T epochs, as opposed to being performed at every epoch.

Table 1: Computational costs analysis in GPU days with single NVIDIA Tesla V100 on DomainNet.

Algorithm	finetuning↓	generation↓	total↓
DSI [15]	22.44	7.94	30.38
CDGA [11]	0	161.11	161.11
ours	**0**	**2.19**	**2.19**

Thirdly, by incorporating Latent Distillation and the efficient sampling strategy, our model swiftly attains good generalization capabilities. Consequently, the number of training iterations is reduced compared to other methods, further diminishing the quantity of generated samples. As for the generation parameters, both DSI [15] and CDGA [11] employ 50 denoising steps. In contrast, our method utilizes a smaller number of denoising steps, thereby reducing the generation time per individual image. The reason for using fewer denoising steps is twofold. Firstly, it allows for the utilization of the latent knowledge in LDM through Latent Distillation. Secondly, a sampling strategy is employed to decompose the styles and contents of the candidates. This approach enables us to achieve satisfactory DG performance without placing excessively high demands on the quality of samples generated by LDM. As a result, fewer denoising steps are employed to further reduce computational costs and improve efficiency. Consequently, our method exhibits significantly lower generation time compared to the previous two methods.

For the training time, it is challenging to make a fair comparison as the first two methods lack sufficiently open-source DG training code. As for our approach, we employ a relatively fixed batch size and number of epochs. Additionally, we only sample one image from every three generated images for augmentation. Consequently, we typically complete all training processes, excluding generation, within 0.4 GPU days using a single NVIDIA Tesla V100. If parallelized across multiple GPUs, obtaining results within several hours is feasible. Therefore, our computational costs are entirely affordable.

4 Experimental Settings

4.1 Settings and Datasets

Following DomainBed [4], we conducted a series of experiments on five prominent real-world benchmark datasets: PACS [7], VLCS [3], OfficeHome [14], TerraIncognita [1], and DomainNet [10]. To ensure a fair and consistent comparison, we follow DomainBed's [4] established training and evaluation protocol. In this protocol, we designate one domain as the target, while the remaining domains serve as source domains. The performance of domain generalization is evaluated individually on each domain and then averaged across all domains. Model selection is conducted using the training-domain validation approach, where 20% of the source domain data is used for validation.

4.2 Hyperparameter tuning

Algorithm-Specific Hyperparameters. In DomainFusion, our algorithm-specific hyperparameters can be categorized into several components. Firstly, we have the generation parameters for Stable Diffusion, namely the guidance scale, and guidance strength. For these parameters, we adopt the common practice used in Stable Diffusion pipelines and employ the default values without conducting any tuning. The second component comprises factors such as the candidate number and loss weights. Due to limitations in computational resources, we only performed a search for the candidate number within the range of [1, 3, 5]. Other hyperparameters related to stable diffusion were not subjected to tuning in our experiments. Hence, in theory, there is untapped potential for further enhancing the performance of our algorithm.
Algorithm-Agnostic Hyperparameters. In DomainFusion, the algorithm-agnostic hyperparameters we employ primarily consist of learning parameters, including batch size, learning rate, weight decay, and the number of epochs. For the learning rate, we searched within the range of [1e-4, 3e-4, 5e-4, 7e-4, 9e-4, 1e-3]. As for the other parameters, we generally used default values, with a batch size of 16, weight decay of 5e-4, and 120 epochs. In theory, by exploring a wider range of hyperparameter values and conducting more comprehensive optimization, it is possible to unlock better results and maximize the algorithm's capabilities.

4.3 Implementation Details

For the latent diffusion model, we employ the stable diffusion v1-4 model. Specifically, we utilize the image-to-image pipeline for image generation and loss extraction, where the input image size is set to 320×320, which greatly boosts algorithm training speed and reduces computational overhead, and other hyperparameters are set to their default values as specified by stable diffusion. For domain generalization, we utilize ResNet-50 pre-trained on ImageNet and RegNet-Y-16GF pre-trained using SWAG as our backbone models. We employ the Adam optimizer and cosine learning rate schedule during training. For algorithm-specific parameters, the candidate number $N = 3$, the interval of generation epochs $T = 5$. The three coefficients of loss are $\lambda_1 = 1$, $\lambda_2 = 0.5$, and $\lambda_3 = 0.5$, which are decided based on empirical observations and the scale of the loss. Moreover, λ is 0.4 for c^*.

5 Additional Analysis and Discussion

5.1 Other Ablation Study

Effects of the Sampling Strategy. Tab. 2 demonstrates the effect of the sampling strategy. Including the sampling strategy led to a significant enhancement of 1.7% in accuracy compared to the exclusion version, thereby indicating the effectiveness of the sampling strategy.

Table 2: Effects of the Sampling Strategy and Synthetic Dataset Updating.

Sampling	Updating	Art	Clipart	Product	Real	Avg.
✗	✓	79.4	71.8	87.5	88.2	81.7
✓	✗	76.7	68.9	85.4	87.2	79.6
✓	✓	**81.2**	**73.9**	**88.5**	**90.1**	**83.4**

Table 3: Effects of the Candidate Number.

Candidate Number	Art	Clipart	Product	Real	Avg.
$N = 1$	79.4	71.8	87.5	88.2	81.7
$N = 3$	**81.2**	**73.9**	**88.5**	**90.1**	**83.4**
$N = 5$	80.4	72.7	87.8	89.3	82.6

Effects of the Candidate Number. Tab. 3 presents the impact of the number of candidates, denoted as N, on the results. We considered $N = 1$, and $N = 5$. In the implementation, N is primarily adjusted by the number of images generated for each prompt in the stable diffusion pipeline. Generally, utilizing more than one candidate tends to yield better results compared to using a single candidate.
Effects of Synthetic Dataset Updating in Augmentation. We also provide an ablation study on the effect of synthetic dataset updating in our augmentation, where we only generate a fixed synthetic dataset on the first epoch, instead of updating it in every generation epoch. As is shown in Tab. 2, The updating of the synthetic dataset resulted in a 2.8% improvement in the DG performance, thereby validating its efficacy.

5.2 More Visualization Results

Visualization of Generated Samples and LD Noise. We present more visualization results of the evolution of generated samples and their corresponding LD noise images in Fig. 3. It can be observed that as the synthetic dataset is continuously updated, the generated samples exhibit cross-domain phenomena. This allows for the retention of a certain degree of content similarity while introducing new styles, thereby serving as evidence of the effectiveness of our method. Moreover, the corresponding LD noise has the ability to disregard domain-specific features such as backgrounds, demonstrating the strong generalization capability of LD noise. This can be observed more clearly in Fig. 4.
Visualization of LD Noise on images with complex backgrounds. To further validate the generalization capability of LD noise, we selected samples with complex backgrounds and observed the performance of LD noise under the influence of additional domain-specific feature interference. As illustrated in Fig. 4, for images with more complex backgrounds, LD noise patterns can also extract semantic information while disregarding the background, further

Table 4: Comparison with other corruptions in office-home.

Corruption Type	Art	Clipart	Product	Real	Avg.
Ours w/o LD	73.6	71.2	80.7	88.7	78.6
Gaussian blur	80.9	72.6	89.5	89.5	83.1
Pepper noise	78.3	72.4	87.1	88.9	81.7
Random mask	77.0	68.2	82.9	86.2	78.6
Ours	81.2	73.9	88.5	90.1	83.4

confirming the effectiveness of the LD method for providing useful information to benefit DG tasks.

5.3 Comparison with other corruption methods.

To further illustrate the effectiveness of our corruption of reducing contrast in Latent Distillation, we also compare it with some other corruption methods like blurring and noise and random masking. Experimental results demonstrate that our corruption is more effective than other corruptions, as shown in Tab. 4. Moreover, we find that our corruption preserves structure information and extracts useful LDM prior, while random masking destroys such information and undermines the LDM prior, as demonstrated by visualization in Fig. 2.

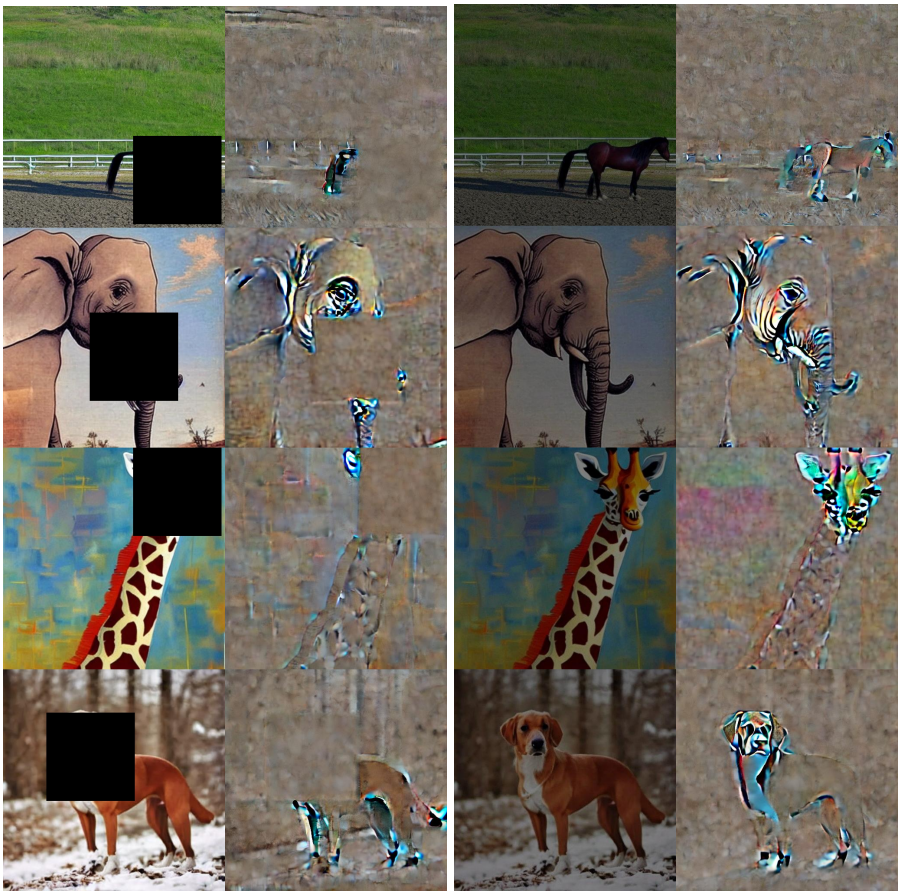
6 Comparison with Diffusion Classifier

In addition to diffusion-based DG methods, we have also observed the existence of some diffusion-based classification methods [2,6]. To provide a comprehensive demonstration of the effectiveness of our approach, we also compared our method with these diffusion-based classification methods as below.

A Recap of Diffusion Classifier. For image classification tasks, the fundamental requirement is to compute the log-likelihood over class labels $\{y_i\}$. Unfortunately, diffusion models do not produce exact log-likelihoods(i.e. directly computing $\log p_\phi(\mathbf{x} \mid y = \mathrm{y}_i)$ is intractable) [5]. However, recent research [2, 6] has provided compelling evidence that $\log p_\phi(\mathbf{x} \mid y = \mathrm{y}_i)$ can be estimated using $\mathcal{L}_{\text{Diff}}$. This is attributed to a profound interrelation between the log-likelihood over its variational lower bound (ELBO) and $\mathcal{L}_{\text{Diff}}$. Specifically, the relationship can be articulated as follows:

$$\begin{aligned} \log p_\phi(\mathbf{x} \mid y) &\geq \text{ELBO} \\ &\approx -\mathbb{E}_{t,\boldsymbol{\epsilon}}\left[w(t)\left\|\hat{\boldsymbol{\epsilon}}_\phi(\mathbf{x}_t; y; t) - \boldsymbol{\epsilon}\right\|_2^2\right] + C \\ &= -\mathcal{L}_{\text{Diff}} + C \end{aligned} \tag{1}$$

where C is a constant independent of the class labels y. Therefore, $-\mathcal{L}_{\text{Diff}}$ can be employed as a proxy to estimate the log-likelihood over class labels



(a) Random masking image and LDM prior. **(b)** Contrast reduction image and LDM prior.

Fig. 2: Contrast Reduction preserves structure information

$\log p_\phi(\mathbf{x} \mid y)$ [2,6], thereby transforming diffusion models into a potential classifier.

Limitations of Diffusion Classifier. Current attempts such as diffusion classifier [2, 6] give score vectors based on the denoising loss, but the effectiveness of such approaches is limited as they match the image with all category labels, which includes inaccurate estimations and leads to confusing results. We substantiate this claim in Fig. 5 that latent diffusion can merely present accurate prediction on matched text-image pairs and fails on mismatched scenarios, where we visualize the score vectors of these approaches using cross-attention maps obtained by DAAM [12]. It can be observed that latent diffusion fails to comprehend mismatched image-text pairs, resulting in noisy score estimations.

Experiment Results of Diffusion Classifier. Two existing methods employ latent diffusion models as a classifier, we denote them as Diffusion Classifier [6]

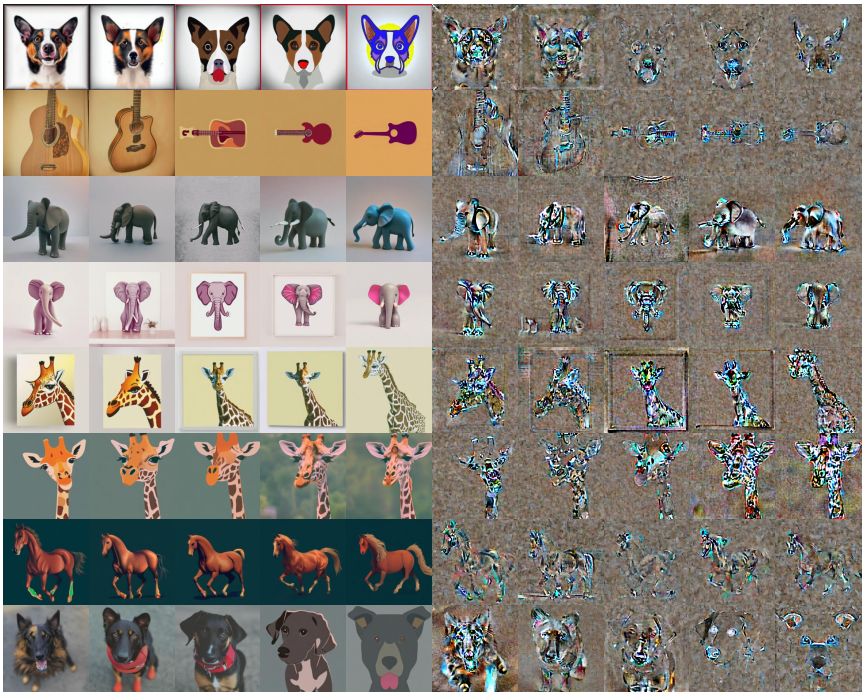


Fig. 3: More visualization results of generated samples and LD noise, with the left section being the evolution of generated samples and the right section being corresponding LD noise.

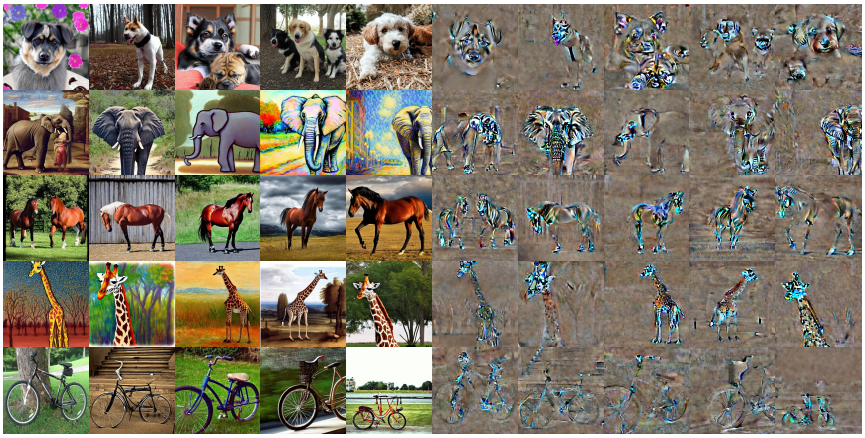


Fig. 4: More visualization results of LD noise on images with complex backgrounds.

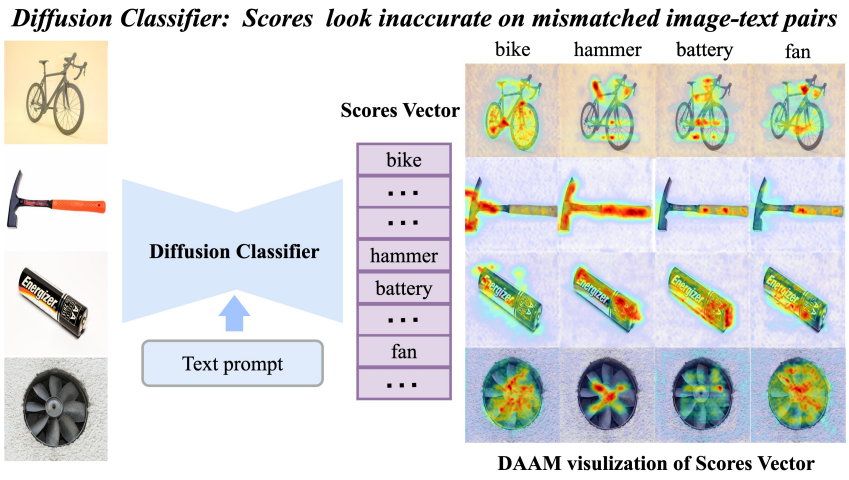


Fig. 5: DAAM visualization of Diffusion Classifier.

Table 5: Comparison with DG methods. The DG accuracy on five domain generalization benchmarks is presented with the best results highlighted in bold. The results denoted by † correspond to the reported numbers from DomainBed [4].

Algorithm	PACS	VLCS	OfficeHome	TerraInc	DomainNet	Avg.
Diffusion Classifier [13]	47.0	40.6	26.8	13.5	10.8	27.7
DiffusionNet [8]	23.8	0.8	15.5	8.5	0.3	9.8
ERM† [13]	85.5	77.5	66.5	46.1	40.9	63.3
Ours	**90.0**	**79.2**	**72.4**	**51.1**	**44.6**	**67.5**

and DiffusionNet [2] respectively. Therefore, we also conduct experiments to compare DomainFusion with them. To ensure fairness, we constrain all three approaches to use stable diffusion and employ the same latent diffusion model parameters. The experimental results are presented in Tab. 5. The findings indicate that both the Diffusion Classifier and DiffusionNet do not exhibit high performance as DG image classifiers.

References

1. Beery, S., Van Horn, G., Perona, P.: Recognition in terra incognita. In: Proceedings of the European conference on computer vision (ECCV). pp. 456–473 (2018)
2. Clark, K., Jaini, P.: Text-to-image diffusion models are zero-shot classifiers. arXiv preprint arXiv:2303.15233 (2023)
3. Fang, C., Xu, Y., Rockmore, D.N.: Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1657–1664 (2013)

4. Gulrajani, I., Lopez-Paz, D.: In search of lost domain generalization. arXiv preprint arXiv:2007.01434 (2020)
5. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems **33**, 6840–6851 (2020)
6. Li, A.C., Prabhudesai, M., Duggal, S., Brown, E., Pathak, D.: Your diffusion model is secretly a zero-shot classifier. arXiv preprint arXiv:2303.16203 (2023)
7. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE international conference on computer vision. pp. 5542–5550 (2017)
8. Li, H., e.a.: Domain generalization with adversarial feature learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5400–5409 (2018)
9. Mordvintsev, A., Pezzotti, N., Schubert, L., Olah, C.: Differentiable image parameterizations. Distill **3**(7), e12 (2018)
10. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 1406–1415 (2019)
11. S., H., et al.: Cross domain generative augmentation: Domain generalization with latent diffusion models. arXiv:2312.05387 (2023)
12. Tang, R., Pandey, A., Jiang, Z., Yang, G., Kumar, K., Lin, J., Ture, F.: What the daam: Interpreting stable diffusion using cross attention. arXiv preprint arXiv:2210.04885 (2022)
13. Vapnik, V.N.: Statistical Learning Theory. Wiley-Interscience (1998)
14. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5018–5027 (2017)
15. Yu, R., Liu, S., Yang, X., Wang, X.: Distribution shift inversion for out-of-distribution prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3592–3602 (2023)