Do Domain Generalization Methods Generalize Well?

Akshay Mehra[1], Bhavya Kailkhura[2], Pin-Yu Chen[3] and Jihun Hamm[1]
[1]Tulane University [2]Lawrence Livermore National Laboratory [3]IBM Research
{amehra, jhamm3}@tulane.edu, kailkhura1@llnl.gov, pin-yu.chen@ibm.com

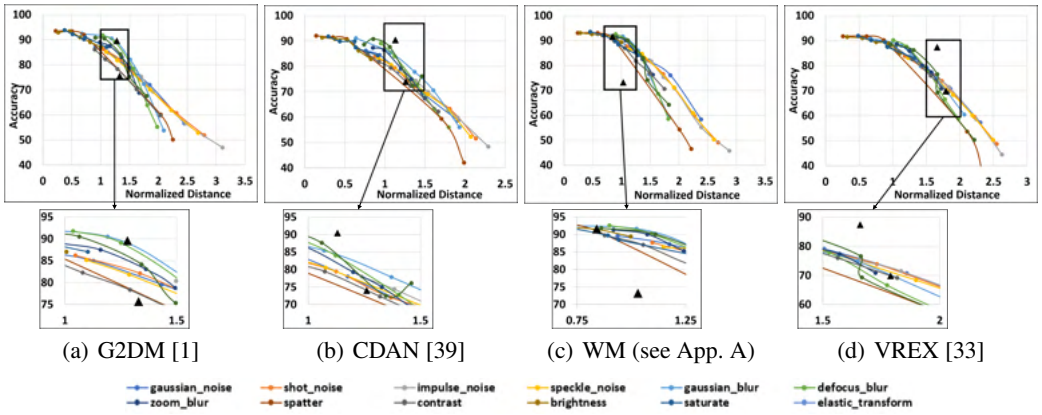
Abstract

Domain Generalization (DG) methods use data from multiple related source domains to learn models whose performance does not degrade on unseen domains at test time. Many DG algorithms rely on reducing the divergence between the source distributions in a representation space to potentially align unseen domains close to the sources. These algorithms are motivated by the analytical works that explain generalization to unseen domains based on their distributional distance (e.g., Wasserstein distance) to the sources. However, we show that the accuracy of a DG model varies significantly on unseen domains equidistant from the sources in the learned representation space. This makes it hard to gauge the generalization performance of DG models only based on their performance on benchmark datasets. Thus, we study the worst-case loss of a DG model at a particular distance from the sources and propose an evaluation methodology based on distributionally robust optimization that efficiently computes the worst-case loss on all distributions within a Wasserstein ball around the sources. Our results show that models trained with popular DG methods incur a high worst-case loss even close to the sources which show their lack of generalization to unseen domains. Moreover, we observe a large gap between the worst-case and the empirical losses of distributions at the same distance, showing the performance of the DG models on benchmark datasets is not representative of their performance on unseen domains. Thus, our (target) data-independent and worst-case loss-based methodology highlights the poor generalization performance of current DG models and provides insights beyond empirical evaluation on benchmark datasets for improving these models.

1 Introduction

Domain Generalization (DG) [63] studies the crucial problem of developing models whose performance remains high on unseen domains (or variations). Analyses in recent works have demonstrated that the distributional distance between the source and the unseen domains is a key metric to predict the generalization performance of the model on unseen domains [56, 34, 23]. To ensure that the unseen domain lies close to the source domains some works in DG impose additional assumptions on the unseen domains whereas others learn a representation space in which the divergence between multiple source domains can be reduced while ensuring high performance on these domains [1, 21, 71] (see App. A and E for a review of DG). By learning such a representation space, these methods aim to potentially reduce the divergence between source and unseen domains, thereby guaranteeing generalization. However, we find that performance of DG models varies substantially even on unseen domains lying at the same distance. In particular, Fig. 1 shows the performance of DG models (trained with Cartoon and Sketch from PACS) on unseen domains versus their distance from the sources in the representation space learned by DG methods. We use Arts and Photos from PACS [38] and corruptions [27, 46] (with 5 severity levels) of the source domains as unseen domains. For individual corruption types, while the distance between the source and unseen domains (in the representation space) correlates well with the error of the DG model, there is high variability in the accuracy of the models on unseen domains at any particular distance. Thus, the performance of DG models on a few

ML Safety Workshop, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) G2DM [1] (b) CDAN [39] (c) WM (see App. A) (d) VREX [33]

Figure 1: (Best viewed in color.) The high variability in the accuracy at the same distance from the sources demonstrates that the performance of a DG method on a few domains is not representative of their performance on other unseen domains. The triangles denote unseen benchmark distributions (Photos and Art for PACS [38] (left to right)) and lines denote distributions under common corruptions.

benchmark datasets is not enough to understand their generalization performance on unseen domains, even those lying at the same distance from the sources.

Thus, in this work we study the worst-case performance of the DG models on all unseen domains lying at a particular distance from the source distributions and propose an evaluation methodology that computes the worst-case performance efficiently. Our methodology, based on distributionally robust optimization (DRO) works by considering the space of probability distributions with Wasserstein distance as a metric, and efficiently computes the loss of the worst-case distribution in a ball around the source distributions. Since DG methods learn a representation space [61] in which realistic unseen domains lie close to the sources (see Fig. 1), we define the Wasserstein ball in this representation space rather than in the input space. Results of our evaluation method show models trained with state-of-the-art DG methods incur high worst-case loss even on distributions lying close to the source distributions in the representation space. This highlights the existence of realistic unseen domains which if encountered at test time can lead to a significant deterioration in the performance of DG models. Moreover, we observe a large gap between the worst-case and empirical losses on distributions at the same distance from the source distributions. This suggests that empirically evaluating the performance of DG methods on a few benchmark datasets cannot effectively gauge their performance on unseen distributions encountered in the real world. Based on our results, we believe that using our evaluation method along with the evaluation on benchmark datasets is necessary for understanding the performance of DG models on unseen domains. Additionally, using insights from our evaluation methodology, we propose an iterative training procedure that minimizes the worst-case loss of DG models at different distances while minimizing specific losses for a DG method. Our algorithm significantly improves the worst-case loss of the DG models with a minor decrease in the performance on source distributions. Thus, our evaluation and training methods are efficient and effective ways for evaluating and improving the generalization of DG models.

2 A distance-based evaluation method for domain generalization

Notation: Let $\mathcal{X}$, $\mathcal{Y}$ denote the data domain and labels, $P_{S/T}(x, y)$ be the joint distributions of features and labels for the source/target domain, and $\mathcal{D}_{S/T}$ denotes examples drawn from $P_{S/T}(x, y)$. For multiple sources, $P_S^i(x, y)$ is i^{th} source. Let $g : \mathcal{X} \to \mathcal{Z}$ be the representation map of an input x to its features in the representation space $\mathcal{Z}$ with parameters θ and let $h : \mathcal{Z} \to \mathcal{Y}$ be the classifier on top of $\mathcal{Z}$ with parameters ζ. A distribution on $\mathcal{Z} \times \mathcal{Y}$ can be a new distribution or a push-forward distribution ($g_\sharp \mathcal{P}$) of an input-space distribution, distinguishable from the context.

Background on Wasserstein distance: Let $\Pi(P, Q)$, be the space of joint probability distributions with marginal distributions P and Q, and c be the cost of transporting mass from (x_1, y_1) to (x_2, y_2). Then the Optimal Transport distance [60, 48, 2], $OT_c(P, Q) = \inf_{\pi \in \Pi(P,Q)} \mathbb{E}_\pi[c((x_1, y_1), (x_2, y_2))]$. With cost c being the ℓ_2-norm, OT_c distance is also known as the type-2 Wasserstein distance,

$W_2(P,Q) = (\inf_{\pi \in \Pi(P,Q)} \mathbb{E}_{(x_1,y_1),(x_2,y_2)\in\pi}[c((x_1,y_1),(x_2,y_2))^2])^{\frac{1}{2}}$. Following [56, 61], we measure OT_c between distributions in the representation space $\mathcal{Z}$ with the cost $c((z_1,y_1),(z_2,y_2)) = \|z_1 - z_2\|_2^2 + \infty \cdot I[y_1 \neq y_2]$. With this choice of c, we have $OT_c = W_2^2$.

Efficient evaluation of the worst-case loss for DG: To provide a (target) data-independent measure of DG performance, we propose to use the DRO framework and measure the worst-case loss of a DG model at a particular distance [6, 44, 18, 69, 23, 7]. For a space of joint distributions $\mathcal{X} \times \mathcal{Y}$ with type-2 Wasserstein distance as a metric [48], the worst-case loss of a distribution P within the distance ρ from source distribution Q is given by $\sup_{P:W_2(P,Q)\leq\rho} \mathbb{E}_{(x,y)\sim P}[\ell(h(g(x)),y)]$. This (primal) problem is an infinite-dimensional problem over a convex set P, making it difficult to solve. However, the optimal value can be computed through a finite-dimensional dual problem [56, 61]

$$\inf_{\gamma\geq 0}\{\gamma\rho^2 + \mathbb{E}_{(x_0,y_0)\sim Q}[\sup_{x\in\mathcal{X}} \{\ell(h(g(x)),y_0) - \gamma\|g(x_0) - g(x)\|^2\}]\}. \tag{1}$$

The equality of the primal and dual problems (strong duality) has been proven under different assumptions on c, ℓ and whether the nominal distribution Q is continuous, empirical ($\frac{1}{N}\sum_i \delta(z - z_i)$, where N denotes number of points in the empirical distribution) or both [23, 56, 44, 69, 9]. However, due to the high correlation between the error of the DG models and the distance of the unseen domains from the source distributions in the representation space (often the layer before the logit/softmax layer [1, 39]), we solve the DRO problem directly in the representation space i.e., we solve

$$\sup_{P:W_2(P,g_\sharp Q)\leq\rho} \mathbb{E}_P[\ell(h(z),y)] = \inf_{\gamma\geq 0}\{\gamma\rho^2 + \mathbb{E}_{(z_0,y_0)\sim g_\sharp Q}[\sup_{z\in\mathcal{Z}} \{\ell(h(z),y_0) - \gamma\|z_0 - z\|_2^2\}]\}. \tag{2}$$

We use Alg. 1, WC-DG, to compute the worst-case loss of a DG model. The proposed algorithm differs from that of [56, 61] who solved the Lagrangian relaxed penalized version of the problem in Eq. 1 while we solve the non-relaxed problem by minimizing over the dual variable γ in Eq. 2. Since the maximization (over the sample z) is in the representation space (see App. B for a discussion on input space vs representation space), we do not require prohibitive assumptions such as Lipschitzness of $\nabla_z \ell(z;\theta)$ and $\nabla_\theta \ell(z;\theta)$, as made in [56]. Moreover, for the maximization in Eq. 2 to have a finite optimal value that is also computationally easy to find, we do not require the differentiability or even continuity of ℓ [23]. This allows us to use losses such as cross-entropy, hinge, and 0/1-loss in our methodology. To solve Eq. 2, we propose to use SGD for cross-entropy and modified hinge loss. For 0/1-loss the surrogate loss is given in the closed-form (see App. B.1 for the details of different losses). For these losses, the problem in Eq. 2 is a saddle point problem that is convex in the scalar γ and is strongly concave in z (for hinge loss and for cross-entropy loss with $\gamma > \gamma_0$, for some γ_0). Thus, the problem can be solved efficiently by alternating SGD as proposed in Alg. 1. For example, computing the worst-case loss of a model in our experiments at a distance ρ using a sample of 1000 points takes only about a minute on a GPU-enabled machine.

Improving DG by minimizing the worst-case loss: To reduce the worst-case loss of DG models at a particular distance, we use DRO-based training that minimizes ($\min_{\zeta,\theta,\gamma\geq 0}\{\gamma\rho^2 + \mathbb{E}_Q[\sup_{z\in\mathcal{Z}} \{\ell(h(z),y_0) - \gamma\|z - z_0\|_2^2\}]\}$) along with the losses of specific DG methods. Our algorithm, DR-DG, is presented in Alg. 2. DR-DG alternates between generating the samples that approximate the worst-case loss at a given distance by solving Eq. 2 over a mini-batch and then minimizes the loss over these samples along with the objectives of the specific DG algorithm. In practice, the additional objective of generating and training on samples that approximate the loss of the worst-case distribution only adds a small overhead to vanilla DG training. The overall run-time increase is proportional to the number of maximization steps per batch (T_2 in Alg. 2). For T_2=20 on PACS with WM and G2DM the run-time increases by mere $\sim$3 seconds per epoch in our experiments.

3 Experiments

Here we present an empirical comparison of WC-DG and DR-DG. We use DG algorithms (see App. A) that learn a representation space by minimizing the divergence between the source distributions during training. In particular, we use Wasserstein Matching (WM), which reduces the Wasserstein distance between the source distributions in the representation space, and G2DM [1] and CDAN [39] which use discriminators to align the source distributions as well as VREX [33] which reduces the variance in the risks on the sources. We use popular benchmark datasets Rotated MNIST (R-MNIST) [24], PACS [38] and VLCS [19]. Similar to [25], we use a convolutional neural network for R-MNIST and fine-tune a ResNet50 model pre-trained on Imagenet for PACS and VLCS. For all our experiments, we arbitrarily chose two domains for training (0° and 15° for R-MNIST, Cartoon and Sketch for PACS,

and Caltech101 and SUN09 for VLCS), while reserving others for various evaluations. Additional results on WC-DG and DR-DG are in App. C along with experimental details in App. D. Our codes are available at `https://github.com/akshaymehra24/CertifiableDG`.

Performance of DG models on unseen distributions: Here we evaluate the models trained with DG methods using the PACS dataset on unseen domains (see App. C.1 for evaluation on R-MNIST). We use Art and Photo along with the corrupted versions of source domains (Cartoon and Sketch) generated by adding variations considered in Imagenet-C [28] (see App. D.3) with of different severity levels to create additional realistic domains. Results in Figs. 1, show high variability in the accuracy of DG models on distributions lying at similar distances in the representation space from the sources. This variability is also observed on benchmark distributions, e.g., Art and Photo lie at a similar distance but have about 15% performance variation. The results highlight the difficulty of evaluating DG models with only benchmark datasets as high performance on one domain (e.g., Photos) does not imply high performance on other domains even at the same distance.

Worst-case loss of DG models: Here we use WC-DG, Alg. 1, to compute the worst-case loss (using three different loss functions) of the DG models at different distances from the source distribution in the representation space. The high worst-case loss in Fig. 2 (top row), even close to the sources, shows the possibility of unseen distributions deteriorating the performance of the DG models. To ensure that the worst-case loss computed in the representation space is not a significant overestimation of the worst-case loss computed in the input space, we compare the results of approximately solving Eq. 1 (input space) to solving Eq. 2. Fig. 2 shows that the worst-case loss obtained through solving Eq. 2 is not a significant overestimation of the worst-case loss from Eq. 1. Since solving Eq. 1 exactly is computationally challenging and requires prohibitive assumption to guarantee the finiteness of the supremum, we use Eq. 2 which does not require such assumption, is computationally much cheaper and exact. Along with corruptions of the sources, we also plot the losses of the adversarial distribution of the sources in Fig. 3 (in App B.1) in the input and representation space, generated using PGD attack [40] with perturbation constrained point-wise to control the deviation of the distribution from the sources. Since the worst-case distribution considered by WC-DG has points distorted by different amounts, it is a more general worst-case distribution than the adversarial distribution with a fixed point-wise budget, and its loss can be higher. Moreover, Fig. 3 (in App B.1), shows a similar trend in the worst-case loss using different loss functions although the inner maximization was better behaved with the modified hinge loss. Lastly, due to the large gap between the worst-case and empirical losses Fig. 2(top), performance on benchmark datasets is not representative of the performance on unseen domains and we use DR-DG to improve the worst-case loss. Fig. 2(bottom) shows significant improvement in the worst-case loss and reduction in the gap between worst-case loss computed by solving Eq. 2 vs. Eq. 1, all while only marginally degrading the accuracy on source distributions (see App. C.5).

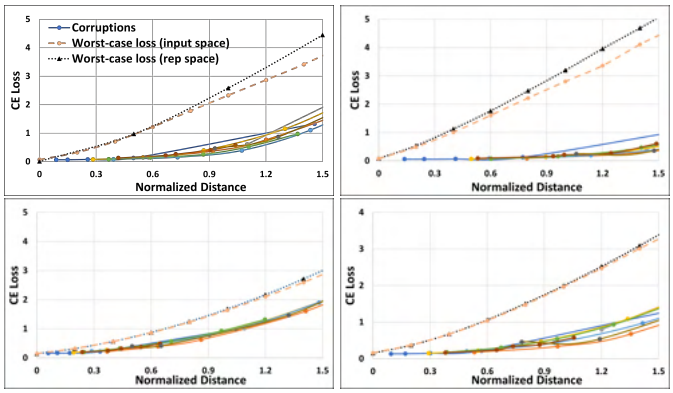


Figure 2: (Best viewed in color.) Comparison of the worst-case loss computed in the representation space (with Eq. 2), in the input space (with Eq. 1) and empirical losses on corrupted domains. Models are trained with WM (left) and G2DM (right) on R-MNIST without DR-DG (top) and with DR-DG (bottom).

4 Conclusion

We showed that the current evaluation of DG methods using a few benchmark datasets is not a good indicator of their performance on unseen domains. Thus, we proposed an evaluation methodology for DG based on DRO that relies on the worst-case loss of a DG model based on the distance in the representation space along with an iterative training procedure that minimizes the worst-case loss of these models on distributions lying within a distance. Our results show that evaluation of DG methods via a data-independent metric such as the worst-case loss provides insights beyond empirical evaluation on benchmark datasets and allows us to objectively judge the progress of DG as a field.

Acknowledgement

This work was supported by the NSF EPSCoR-Louisiana Materials Design Alliance (LAMDA) program #OIA-1946231 and by the LLNL-LDRD Program under Project No. 20-ERD-014. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] Isabela Albuquerque, João Monteiro, Mohammad Darvishi, Tiago H Falk, and Ioannis Mitliagkas. Generalizing to unseen domains via distribution matching. *arXiv preprint arXiv:1911.00804*, 2019.

[2] David Alvarez-Melis and Nicolo Fusi. Geometric dataset distances via optimal transport. *arXiv preprint arXiv:2002.02923*, 2020.

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

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[5] Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137, 2007.

[6] Aharon Ben-Tal, Dick den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

[7] Dimitris Bertsimas, Vishal Gupta, and Nathan Kallus. Data-driven robust optimization. *Mathematical Programming*, 167(2):235–292, 2018.

[8] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24:2178–2186, 2011.

[9] Jose Blanchet and Karthyek Murthy. Quantifying distributional model risk via optimal transport. *Mathematics of Operations Research*, 44(2):565–600, 2019.

[10] Dan A Calian, Florian Stimberg, Olivia Wiles, Sylvestre-Alvise Rebuffi, Andras Gyorgy, Timothy Mann, and Sven Gowal. Defending against image corruptions through adversarial augmentations. *arXiv preprint arXiv:2104.01086*, 2021.

[11] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[12] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[13] Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. *arXiv preprint arXiv:1902.02918*, 2019.

[14] Zac Cranko, Zhan Shi, Xinhua Zhang, Richard Nock, and Simon Kornblith. Generalised lipschitz regularisation equals distributional robustness. In *International Conference on Machine Learning*, pages 2178–2188. PMLR, 2021.

[15] Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 447–463, 2018.

[16] James Diffenderfer, Brian Bartoldson, Shreya Chaganti, Jize Zhang, and Bhavya Kailkhura. A winning hand: Compressing deep networks can improve out-of-distribution robustness. *Advances in Neural Information Processing Systems*, 34, 2021.

[17] Andrea Dittadi, Frederik Träuble, Francesco Locatello, Manuel Wüthrich, Vaibhav Agrawal, Ole Winther, Stefan Bauer, and Bernhard Schölkopf. On the transfer of disentangled representations in realistic settings. *arXiv preprint arXiv:2010.14407*, 2020.

[18] John Duchi, Peter Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv preprint arXiv:1610.03425*, 2016.

[19] Chen Fang, Ye Xu, and Daniel N Rockmore. 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*, pages 1657–1664, 2013.

[20] Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *Journal of Machine Learning Research*, 22(78):1–8, 2021.

[21] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015.

[22] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

[23] Rui Gao and Anton J Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *arXiv preprint arXiv:1604.02199*, 2016.

[24] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pages 2551–2559, 2015.

[25] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[26] Tatsunori Hashimoto, Megha Srivastava, Hongseok Namkoong, and Percy Liang. Fairness without demographics in repeated loss minimization. In *International Conference on Machine Learning*, pages 1929–1938. PMLR, 2018.

[27] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[28] Dan Hendrycks and Thomas G Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. *arXiv preprint arXiv:1807.01697*, 2018.

[29] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[30] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pages 2029–2037. PMLR, 2018.

[31] Fredrik D Johansson, David Sontag, and Rajesh Ranganath. Support and invertibility in domain-invariant representations. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 527–536. PMLR, 2019.

[32] Klim Kireev, Maksym Andriushchenko, and Nicolas Flammarion. On the effectiveness of adversarial training against common corruptions. *arXiv preprint arXiv:2103.02325*, 2021.

[33] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[34] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations research & management science in the age of analytics*, pages 130–166. Informs, 2019.

[35] Aounon Kumar, Alexander Levine, Tom Goldstein, and Soheil Feizi. Certifying model accuracy under distribution shifts. *arXiv preprint arXiv:2201.12440*, 2022.

[36] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

[37] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. In *Advances in Neural Information Processing Systems*, pages 9464–9474, 2019.

[38] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[39] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31, 2018.

[40] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[41] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[42] Akshay Mehra, Bhavya Kailkhura, Pin-Yu Chen, and Jihun Hamm. How robust are randomized smoothing based defenses to data poisoning? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13244–13253, 2021.

[43] Akshay Mehra, Bhavya Kailkhura, Pin-Yu Chen, and Jihun Hamm. Understanding the limits of unsupervised domain adaptation via data poisoning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[44] Peyman Mohajerin Esfahani and Daniel Kuhn. Data-driven distributionally robust optimization using the wasserstein metric: Performance guarantees and tractable reformulations. *Mathematical Programming*, 171(1):115–166, 2018.

[45] Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2020.

[46] Norman Mu and Justin Gilmer. Mnist-c: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*, 2019.

[47] Nicolas Papernot, Fartash Faghri, Nicholas Carlini, Ian Goodfellow, Reuben Feinman, Alexey Kurakin, Cihang Xie, Yash Sharma, Tom Brown, Aurko Roy, Alexander Matyasko, Vahid Behzadan, Karen Hambardzumyan, Zhishuai Zhang, Yi-Lin Juang, Zhi Li, Ryan Sheatsley, Abhibhav Garg, Jonathan Uesato, Willi Gierke, Yinpeng Dong, David Berthelot, Paul Hendricks, Jonas Rauber, and Rujun Long. Technical report on the cleverhans v2.1.0 adversarial examples library. *arXiv preprint arXiv:1610.00768*, 2018.

[48] Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[49] Fengchun Qiao, Long Zhao, and Xi Peng. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12556–12565, 2020.

[50] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[51] 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 preprint arXiv:1911.08731*, 2019.

[52] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 11292–11303, 2019.

[53] Vikash Sehwag, Saeed Mahloujifar, Tinashe Handina, Sihui Dai, Chong Xiang, Mung Chiang, and Prateek Mittal. Robust learning meets generative models: Can proxy distributions improve adversarial robustness? *arXiv preprint arXiv:2104.09425*, 2021.

[54] Soroosh Shafieezadeh-Abadeh, Daniel Kuhn, and Peyman Mohajerin Esfahani. Regularization via mass transportation. *Journal of Machine Learning Research*, 20(103):1–68, 2019.

[55] Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. Wasserstein distance guided representation learning for domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[56] Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571*, 2017.

[57] Matthew Staib and Stefanie Jegelka. Distributionally robust optimization and generalization in kernel methods. *Advances in Neural Information Processing Systems*, 32, 2019.

[58] Jiachen Sun, Akshay Mehra, Bhavya Kailkhura, Pin-Yu Chen, Dan Hendrycks, Jihun Hamm, and Z Morley Mao. Certified adversarial defenses meet out-of-distribution corruptions: Benchmarking robustness and simple baselines. *arXiv preprint arXiv:2112.00659*, 2021.

[59] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[60] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[61] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. *Advances in neural information processing systems*, 31, 2018.

[62] Haotao Wang, Chaowei Xiao, Jean Kossaifi, Zhiding Yu, Anima Anandkumar, and Zhangyang Wang. Augmax: Adversarial composition of random augmentations for robust training. *Advances in Neural Information Processing Systems*, 34, 2021.

[63] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*, 2021.

[64] Maurice Weber, Linyi Li, Boxin Wang, Zhikuan Zhao, Bo Li, and Ce Zhang. Certifying out-of-domain generalization for blackbox functions. *arXiv preprint arXiv:2202.01679*, 2022.

[65] Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295. PMLR, 2018.

[66] Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. *arXiv preprint arXiv:2001.02378*, 2020.

[67] Guojun Zhang, Han Zhao, Yaoliang Yu, and Pascal Poupart. Quantifying and improving transferability in domain generalization. *arXiv preprint arXiv:2106.03632*, 2021.

[68] Hanlin Zhang, Yi-Fan Zhang, Weiyang Liu, Adrian Weller, Bernhard Schölkopf, and Eric P Xing. Towards principled disentanglement for domain generalization. *arXiv preprint arXiv:2111.13839*, 2021.

[69] Chaoyue Zhao and Yongpei Guan. Data-driven risk-averse stochastic optimization with wasserstein metric. *Operations Research Letters*, 46(2):262–267, 2018.

[70] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. On learning invariant representations for domain adaptation. In *International Conference on Machine Learning*, pages 7523–7532. PMLR, 2019.

[71] Han Zhao, Shanghang Zhang, Guanhang Wu, José MF Moura, Joao P Costeira, and Geoffrey J Gordon. Adversarial multiple source domain adaptation. *Advances in neural information processing systems*, 31, 2018.

Appendix

We provide a brief review of various DG algorithms used in our paper along with existing analyses of the DG problem in App. A and related work in App. E. We discuss details about the computation of the worst-case loss using different loss functions and the difference between worst-case loss computed in the input-space and representation-space in App. B along with highlighting the difference between point-wise and distributional robustness. We present additional evaluation results of WC-DG and DR-DG in App. C, with details of our experiments and model architectures in App. D.

A Review of DG algorithms and analyses

A.1 DG algorithms

In this section, we review the DG methods used in this work and describe their objective functions. For detailed descriptions of these algorithms, we refer the reader to original works.

G2DM [1]: Let N_S be the number of source domains. This algorithm relies on using N_S one-vs-all discriminators to distinguish a source domain from other $N_S - 1$ source domains. By learning a representation that fools the discriminator this method aims to align the source domains in the representation space. Mathematically, it solves the following problem

$$\min_{g,h} \max_{\tau_1,\cdots,\tau_{N_S}} \mathcal{L}(\mathcal{D}_S; h, g) - \sum_{k=1}^{N_S} \mathcal{L}_k(D_k(g(x)), y_k),$$

where $\mathcal{L}$ denotes the average loss of a given loss ℓ (we used cross-entropy loss) on all points in the sources, D_k denotes the k^{th} discriminator with parameters τ_k, y_k denotes binary labels such that the points belonging to the k^{th} source domain are labeled as zeros and points from all other domains are labeled as ones and $\mathcal{L}_k$ denotes the classification loss of the k^{th} discriminator.

CDAN [39]: Similar to G2DM, this method also utilizes discriminators to align the source distributions. The discriminators used in this approach are conditioned to use the labels of the source domain data for alignment. The overall mathematical objective remains the same as G2DM except for the discriminators in G2DM are replaced by conditional discriminators which use multi-linear conditioning (see [39] for more details). For CDAN, we used binary discriminators which use a one-vs-all strategy similar to G2DM.

Wasserstein Matching (WM): Many previous works have demonstrated the effectiveness of using Wasserstein distance for domain alignment, especially in the domain adaptation literature [55, 15]. Building on that we propose to use Wasserstein matching to reduce the divergence between the source domains in the representation space similar to G2DM and CDAN. Mathematically, our objective still remains the same as G2DM, except that the discriminators are now replaced with Wasserstein distance-based terms which consider alignment in a one-vs-all fashion. Concretely, our objective becomes

$$\min_{g,h} \mathcal{L}(\mathcal{D}_S; h, g) + \sum_{k=1}^{N_S} W_2^2(\mathcal{D}_S^k, \mathcal{D}_S \backslash \mathcal{D}_S^k),$$

where $\mathcal{D}_S \backslash \mathcal{D}_S^k$ denotes all source domains except the k^{th} source domain. To efficiently solve this problem we use the methodology proposed in [15], where the coupling is computed batch-wise using a fixed cost matrix and then this coupling is used to optimize the cost matrix. The dissimilarity cost between points (x, y) and (x', y') used to populate the cost matrix is computed using the feature distance i.e. $\|g(x) - g(x')\|_2^2$ and label distance i.e. $\|y - y'\|_2^2$ (assuming y are one-hot encoded vector of labels). Thus the overall dissimilarity cost between two points (x, y) and (x', y') is $\|g(x) - g(x')\|_2^2 + \lambda\|y - y'\|_2^2$. In our experiments we set $\lambda = 1$.

VREX [33]: This work proposed to use the variance of the risks as a regularizer (VREX) to learn models which can generalize well to unseen domains. In particular, they proposed adding an additional objective to the average error of the source domains, thereby solving

$$\min_{g,h} \mathcal{L}(\mathcal{D}_S; h, g) + \beta \text{Var}(\{\mathcal{L}(\mathcal{D}_S^1; h, g), \cdots, \mathcal{L}(\mathcal{D}_S^{N_S}; h, g)\}).$$

In our experiments, we fix the parameter β to be 1.

Algorithm 1 WC-DG: Evaluation of the Worst-Case loss for a Domain Generalization model

Input: Radius ρ, Representation map g_θ, Classifier h_ζ, Source data $\mathcal{D}_S$, $T_1, T_2 = 1$, α, β, $\gamma_{init}, \gamma_{min}, \gamma_{max}$.
Output: l_{worst} (worst-case loss within a ball of radius ρ)

Init: $\gamma \leftarrow \gamma_{init}$, $\mathcal{D} \leftarrow (g_\theta(\mathcal{X}_S), \mathcal{Y}_S)$
for $m = 1, \cdots, T_1$ **do**
  for a batch $(x, y) \sim \mathcal{D}_S$ and corresponding batch $z \sim \mathcal{D}$ of size n **do**
    for $t = 1, \cdots, T_2$ **do**
      for $i = 1, \cdots, n$ **do** ▷ Solve the inner maximization over z
        $\phi_\gamma \leftarrow \ell(h_\zeta(z^i), y^i) - \gamma \|z^i - g_\theta(x^i)\|_2^2$
        $z^i \leftarrow z^i + \alpha \nabla_z \phi_\gamma$ ▷ Update z and store back in $\mathcal{D}$
      $\gamma \leftarrow \gamma - \beta\{\rho^2 - \frac{1}{n}\sum_{i=1}^n \|z^i - g_\theta(x^i)\|_2^2\}$ ▷ Update γ using the gradient
      $\gamma \leftarrow \text{Clip}(\gamma, \gamma_{min}, \gamma_{max})$ ▷ Clip the value of γ
$\ell_{worst} \leftarrow \{\gamma\rho^2 + \frac{1}{N}\sum_{(x,y)\sim\mathcal{D}_S, z\sim\mathcal{D}}[\phi_\gamma((x, y, z))]\}$ ▷ Return the worst-case loss in the ρ ball

A.2 Analyses of DG using distributional divergence

In this section, we review some of the recent analyses of the DG problem which have demonstrated that generalization to an unseen domain can be studied in terms of the distributional distance between the source and the unseen distributions. Please refer to the original works for detailed descriptions of the assumptions and results.

[5], [55]: In one of the early works, it was shown that performance on an unknown target domain can be estimated based on the performance on the source domain and the distance between the marginal distributions and labeling functions of the two domains. Let a domain be (P, f), where P denotes the distribution on inputs $\mathcal{X}$ and $f : \mathcal{X} \to [0, 1]$ denotes the labeling function. Let $h : \mathcal{X} \to \{0, 1\}$ be a hypothesis and $\mathcal{E}_P(h, f) := \mathbb{E}_{x\sim P}[|h(x) - f(x)|]$ denote the risk of the hypothesis h. Then it was shown in Theorem 1 of [5] that

$$\mathcal{E}_T(h, f_T) \leq \mathcal{E}_S(h, f_S) + d_1(P_S, P_T) + \min\{\mathbb{E}_{P_S}[|f_S(x) - f_T(x)|], \mathbb{E}_{\mathcal{D}_T}[|f_S(x) - f_T(x)|]\},$$

where d_1 denotes the total variation distance. [55], showed a similar result (Theorem 1 of [55]) using type-1 Wasserstein distance for all $K-$Lipschitz continuous hypotheses i.e.,

$$\mathcal{E}_T(h, f_T) \leq \mathcal{E}_S(h, f_S) + 2K \cdot W_1(P_S, P_T) + \lambda,$$

where λ is the combined error of the ideal hypothesis h^* that minimizes the combined error $\mathcal{E}_S(h, f_S) + \mathcal{E}_T(h, f_T)$.

[35]: Recent work showed that the accuracy of smoothed classifiers, smoothed using different smoothing functions under distribution shifts can also be bounded using a function dependent on the Wasserstein distance between the source and target domains. Theorem 4.1 [35] shows that for a function $h : \mathcal{X} \times \mathcal{Y} \to [0, 1]$, whose smoothed version is defined as $\overline{h}(x, y) := \mathbb{E}_{x'\sim S(x)}[h(x', y)]$,

$$|\mathbb{E}_{(x_1,y_1)\sim S}[\overline{h}(x_1, y_1)] - \mathbb{E}_{(x_2,y_2)\sim T}[\overline{h}(x_2, y_2)]| \leq \psi(\rho),$$

where ρ is the radius of the Wasserstein ball around S.

[53]: Another recent work showed that distribution divergence can also be used to explain the transfer of robustness across the two domains. In particular, Theorem 1 [53] shows that the difference in the average margin ($Rob(h, P) := \mathbb{E}_{(x,y)\sim P}[\inf_{h(x')\neq y} \|x' - x\|]$) of the classifier $h : \mathcal{X} \to \mathcal{Y}$ under distribution shift is bounded by conditional Wasserstein distance between the two domains (see the original paper for definition).

$$|Rob(h, P_S) - Rob(h, P_T)| \leq W_{cond}(P_S, P_T).$$

Assuming robust error (RE_ϵ) at a perturbation ϵ is defined as the probability of having margin ($mar := \inf_{h(x')\neq y} \|x - x'\|$) less than ϵ i.e., $RE_\epsilon := Pr[mar < \epsilon]$ and average margin $\mathbb{E}[mar]$. Using Markov's inequality, we have $1 - RE_\epsilon \leq \epsilon^{-1}\mathbb{E}[mar]$. Since $1 - RE_\epsilon = RA_\epsilon$, the robust accuracy at ϵ, we get a bound on the robust error under distribution shift, as follows

$$|RE_\epsilon(P_S) - RE_\epsilon(P_T)| \leq \epsilon^{-1} W_{cond}(P_S, P_T).$$

Algorithm 2 DR-DG: Distributionally-Robust Domain Generalization

Input: $F, T_1, T_2, \alpha, \beta, \eta, \gamma_{init}, \gamma_{min}, \gamma_{max}, \mathcal{D}_S = \{\mathcal{D}_S^j\}_{j=1}^{N_S}, \mathcal{D}_S^j = \{(x_i^j, y_i^j)\}_{i=1}^{N_S^j}, \ell_{dg}$
Output: Representation model parameters θ, Classifier parameters ζ.

Init: Initialize parameters of the representation g_θ and classification h_ζ models.
Init: $\gamma \leftarrow \gamma_{init}, \mathcal{D} \leftarrow g_\theta(\mathcal{X}_S)$
for $m = 1, \cdots, T_1$ **do**
  $\mathcal{D}_{adv} \leftarrow \mathrm{GenAdvDist}(\mathcal{D}_S, g_\theta, h_\zeta)$ ▷ Generate adversarial distribution via CW-attack [11]
  $\rho \leftarrow$F$\cdot W_2(\mathcal{D}_{adv}, (g_\theta(\mathcal{X}_S), \mathcal{Y}_S))$ ▷ Initialize the ball size ρ
  for a batch $(x, y) \sim \mathcal{D}_\mathrm{S}$ and corresponding batch $z \sim \mathcal{D}$ of size n **do**
    for $t = 1, \cdots, T_2$ **do**
      for $i = 1, \cdots, n$ **do** ▷ Solve the maximization over z
        $\phi_\gamma \leftarrow \ell(h_\zeta(z^i), y^i) - \gamma\|z^i - g_\theta(x^i)\|_2^2$
        $z^i \leftarrow z^i + \alpha \nabla_z \phi_\gamma$ ▷ Update and store z
      $\gamma \leftarrow \gamma - \beta\{\rho^2 - \frac{1}{n}\sum_{i=1}^n \|z^i - g_\theta(x^i)\|_2^2\}$ ▷ Update γ using the gradient
      $\gamma \leftarrow \mathrm{Clip}(\gamma, \gamma_{min}, \gamma_{max})$ ▷ Clip the value of γ

    $\ell_{dro} \leftarrow \frac{1}{n}\sum_{i=1}^n \ell(h_\zeta(z^i), y^i)$ ▷ Loss on the worst-case distribution in ρ-ball
    $\ell_{total} \leftarrow \ell_{dro} + \ell_{dg}$ ▷ ℓ_{dg} is the loss of a given DG method.
    $\theta \leftarrow \theta - \eta \nabla_\theta \ell_{total}$ ▷ Update Representation model params
    $\zeta \leftarrow \zeta - \eta \nabla_\zeta \ell_{total}$ ▷ Update Classification model params

Along with these works, a large body of works exists in the DRO literature which explicitly considers their uncertainty set to be based on different divergence measures such as the Wasserstein distance, $f-$divergences such as Jenson Shannon, Kullback–Leibler, Hellinger distance as discussed in the related work in App. E. This highlights the important role of distributional divergence in assessing the performance of domain generalization methods on new unseen domains.

B Additional discussions

In this section, we discuss computing the worst-case loss using different loss functions in WC-DG and also present a comparison of worst-case loss in the input space versus representation space.

B.1 Worst-case loss with three different loss functions

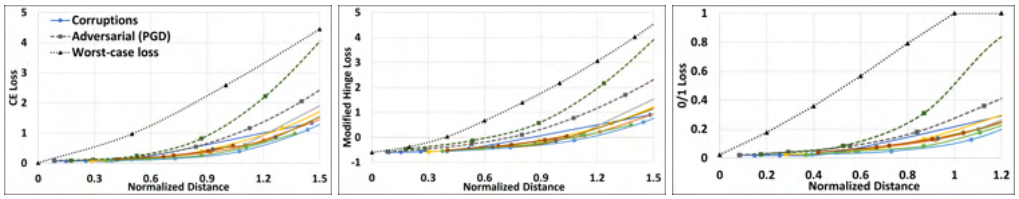
As discussed in the main paper, the proof of strong duality (Eq. 1) has been shown to hold under different assumptions as well as loss functions. Here we show that WC-DG can be used with the hinge loss and the 0/1 loss; the latter can be used to evaluate the accuracy of the DG models.

B.1.1 Hinge loss

When the loss is piece-wise affine $l(z) = \max_{k \le K} [a_k^T z + b_k]$ such as the hinge loss, the dual problem can be written as simple linear programming (Corollary 5.1, Remark 6.6 [44], Remark 9 [23]), and therefore the optimum $\inf_{\gamma \ge 0} \left[\gamma\rho^2 + E[\phi_\gamma(z_0, y_0)]\right]$ can be computed accurately and efficiently. We use SGD to find the optimum for an unconstrained saddle-point problem in our approach. However, the vanilla hinge loss requires modification to be used with SGD, because the loss gradient is zero ($\nabla_z \ell(z) = 0$) for correctly classified point z, and therefore SGD does not progress. We address this by using the modified hinge loss (similar to leaky ReLU)

$$\ell_{\text{modified\_hinge}}(t) = \max\{0,\ 1 - t\} - \alpha \max\{0,\ t - 1\} \tag{3}$$

where t denotes the difference between the logit of the true class and the maximum logit of the other classes i.e., $t = h_y(x) - h_{other}(x)$ for a point (x, y) and α is a small constant such as 0.1. Different from the vanilla hinge loss, the modified loss can have negative values for correctly classified points. This may be undesirable when the objective is loss minimization but has no impact when the objective is loss maximization as is required in WR-DG.



(a) CE loss (b) Modified hinge loss (c) 0/1 loss

Figure 3: (Best viewed in color.) Comparison of worst-case loss and empirical losses on corrupted and adversarial domains of a model trained with WM on R-MNIST. The models trained with Vanilla WM on R-MNIST are evaluated using WC-DG with three loss functions.

B.1.2 0/1 loss

Here we describe how our evaluation method can be used to evaluate the accuracy of the models trained with DG methods. We consider 0/1 loss for evaluating accuracy. Using the 0/1 loss, the solution to the robust surrogate loss, Eq. 2, can be computed in closed form as shown below:

$$\sup_{z} \{\ell(z) - \gamma c(z, z_0)\} = \max\{0,\ 1 - \gamma \|z_0 - z_{adv}\|_2^2\}, \tag{4}$$

where (z_0, y_0) is the initial point and $z_{adv} = \arg\min_z \ \|z - z_0\|_2^2 \ \text{ s.t. } \ h(z) \neq y_0$. Therefore the objective of the evaluation is

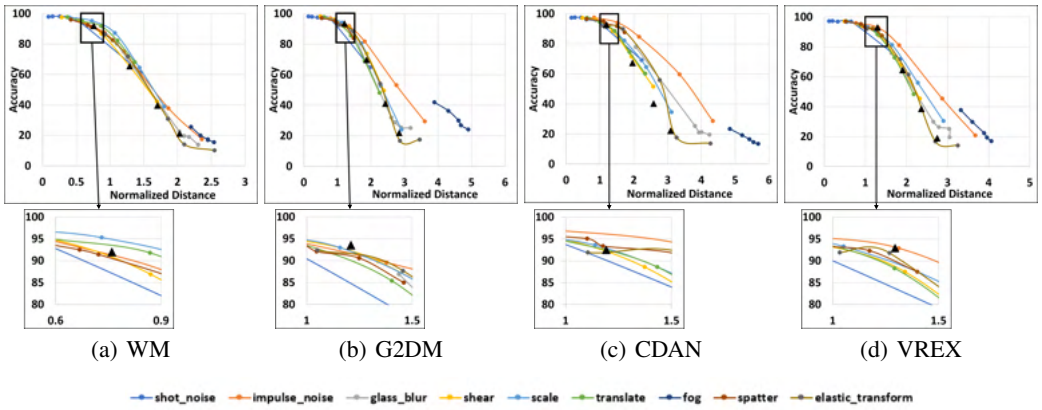
$$\inf_{\gamma \geq 0} \left\{ \gamma \rho^2 + \frac{1}{N} \sum_i \max\{0,\ 1 - \gamma \|z_0^i - z_{adv}^i\|\} \right\}. \tag{5}$$

The objective function is therefore a non-negative, piece-wise linear, non-continuous, and convex function of γ. The non-continuity of the function prevents us from using SGD to optimize over γ. However, minimization over γ is just a one-dimensional convex optimization even though it's non-differentiable. This problem can be efficiently solved using simple scalar minimization methods (such as "scipy.optimize.minimize_scalar") based on the bisection method to solve for the optimal γ. For practically solving the evaluation problem we compute z_{adv} using CW attack [12] and then solve the minimization problem over γ using the $minimize\_scalar$ function with the bounded solver and bound the value of γ to be in [1E-20, 100]. We present evaluation results on a model trained with WM on R-MNIST in Fig. 3(c). The worst-case loss at a normalized distance of 1 is the highest since our definition of normalized distance uses the adversarial distribution which is the nearest distribution (to the sources in the representation space) with zero accuracies (see App.D for details on normalized distance).

While our method can be used for evaluating the model performance with different loss functions such as the cross-entropy loss, hinge loss, or the misclassification (0/1) loss efficiently, we consider only cross-entropy loss for DR-DG. This choice is due to the differentiability of the worst-case loss and also the practical observations that most deep learning models perform better when trained with the cross-entropy loss compared to other losses such as hinge loss.

B.2 Comparison of the worst-case loss in the input-space vs representation-space

In this section, we discuss the differences in evaluating the worst-case loss considering the Wasserstein-ball in the input space (Eq. 1) versus in the representation space (Eq. 2) and discuss why evaluation in the representation space is easier both from a theoretical and a practical standpoint. Measuring worst-case loss when distance between the distributions is measured in the input space is not useful since the distributional distance between the source distributions and the unseen distributions can be large (i.e., when $c((x_0, y_0), (x_1, y_1)) = \|x_0 - x_1\|_2^2 + \infty \cdot I[y_0 \neq y_1]$ in the Wasserstein distance). For example, many previous works have shown that DG methods perform reasonably well on the R-MNIST dataset. But, we observe that the Wasserstein distance measured in the input space between the source distributions and the well-performing target distribution is very large. In particular, models trained with WM on rotation angles $0^\circ, 15^\circ, 30^\circ, 45^\circ$, and 60° yield $\sim$95% accuracy on a domain with rotation angle 75°. But the normalized distance of the 75° domain from the sources is roughly 5, which is too large a radius for the Wasserstein ball to provide any meaningful guarantees. Thus, the



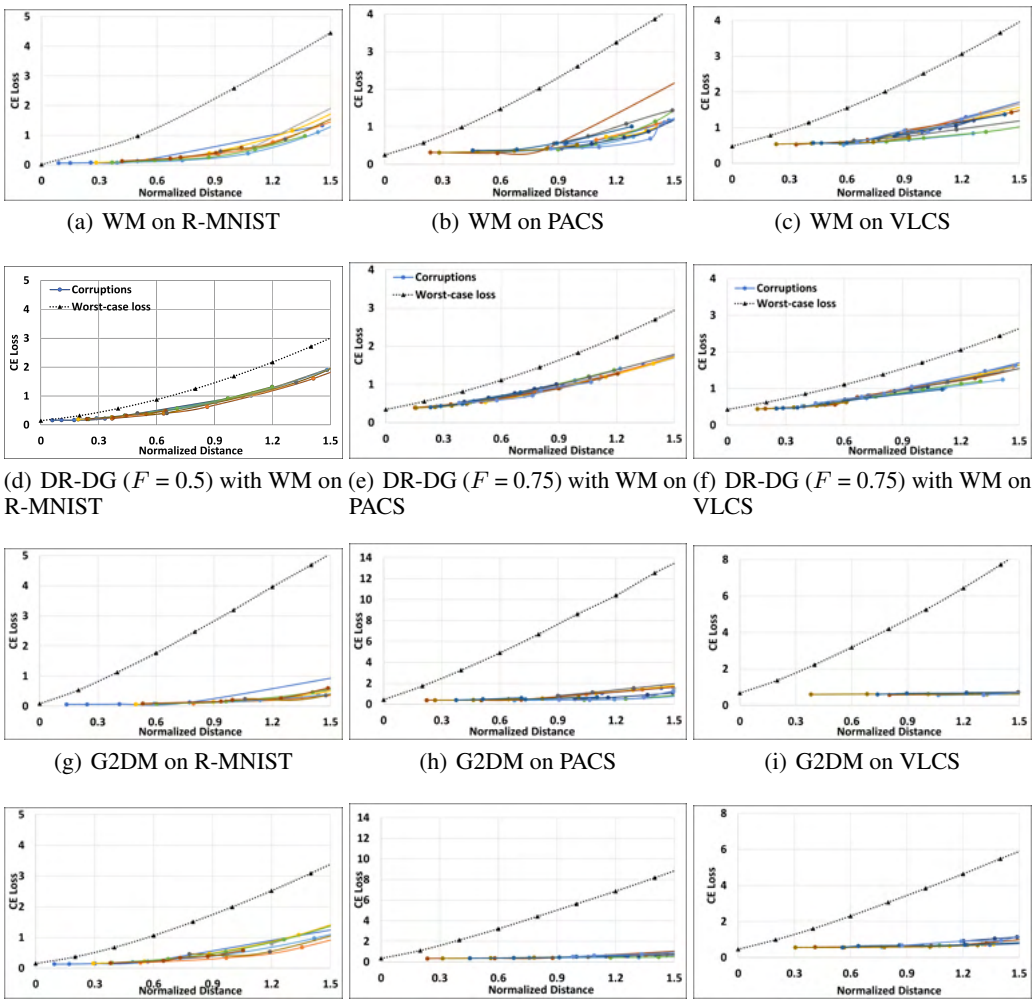
(a) WM (b) G2DM (c) CDAN (d) VREX

Figure 4: (Best viewed in color.) **Difficulty of empirically evaluating the generalization performance of a DG method:** The difference in the performance of DG methods on data from different unseen domains demonstrates that comparing the performance of DG methods using a few benchmark datasets is not sufficient for assessing the generalization performance of DG methods. The triangles denote unseen benchmark distributions ($30°, 45°, 60°, 75°$ for R-MNIST (left to right)) and lines denote distributions under common corruptions with different severity levels.

high performance of DG methods this far from the sources does not give us meaningful information about the performance of the method on unseen distributions lying at that distance. In contrast, the representation space learned by DG methods is much more useful for evaluation since the DG methods explicitly reduce the distance between the distributions by either using Wasserstein matching or discriminator-based approaches or DR-DG. Although there are distinct trade-offs between different methods of alignment, unseen distributions in general lie much closer to the sources compared to their distances in the input space. As long as DG algorithms can bring unseen domains close to the sources in the representation space, evaluation using the representation-space distance is preferable compared to the input-space distance.

A major advantage of computing the worst-case loss by considering the Wasserstein ball in the representation space is the ease of solving the inner maximization in Eq. 2 compared to solving the version in Eq. 1, which considers the ball in the input space. As discussed extensively in [56], finding the maximizer of the inner maximization is in general an NP-hard problem for networks with ReLU activations. Even if the ReLU activations are changed to smoother ones, there are still problems with provably maximizing the inner objective and has been shown to be possible only when additional assumptions such as bounded Lipschitz gradient w.r.t to the input and the model parameters are imposed. These assumptions are prohibitive and restrict evaluation only for small to medium size problems. However, when evaluating in the representation space, we don't need these assumptions since representation space is usually followed by a single softmax layer, and maximization of the inner objective is much easier both theoretically and practically. In this work we restrict the evaluation only in the second to last layer. Although evaluation can be applied to other layers but additional assumptions are needed to guarantee convergence of the evaluation procedure. Some of these assumptions, as discussed earlier, could be prohibitive for large neural networks or may require changes to the architecture. Moreover, as we consider layers closer to the input, it becomes more difficult for DG algorithms to reduce the distance between the sources, thereby leading to higher worst-case loss.

A potential disadvantage of our evaluation method is that it considers the uncertainty set to be distributions in the representation space. This may overestimate the worst-case loss since this space can have a worst-case distribution that is not a push-forward distribution, i.e., one that is not generated through the representation map. Considering only push-forward distributions is possible by adding additional constraints to our uncertainty set but it significantly increases the computational complexity of both WC-DG and DR-DG since the maximizer of the inner objective must be computed through the representation map (Eq. 1). Although solving Eq. 1, restricts the worst-case distribution to be the push-forward distribution, we find the worst-case loss to be very similar in both cases, as shown in Fig. 2. The gap between the worst-case losses becomes even smaller when the models are



(a) WM on R-MNIST (b) WM on PACS (c) WM on VLCS

(d) DR-DG (F = 0.5) with WM on R-MNIST (e) DR-DG (F = 0.75) with WM on PACS (f) DR-DG (F = 0.75) with WM on VLCS

(g) G2DM on R-MNIST (h) G2DM on PACS (i) G2DM on VLCS

(j) DR-DG (F = 1.5) with G2DM on R-MNIST (k) DR-DG (F = 0.75) with G2DM on PACS (l) DR-DG (F = 0.4) with G2DM on VLCS

Figure 5: (Best viewed in color.)Worst-case loss of the models trained with WM and G2DM on R-MNIST, PACS, and VLCS datasets. Rows 1 and 3 show models trained with Vanilla DG methods and rows 2 and 4 show models trained with DR-DG using additional losses from WM and G2DM, respectively. The models trained with DR-DG incur smaller worst-case losses compared to their vanilla counterparts and only slightly higher loss on unseen distributions created through common corruptions.

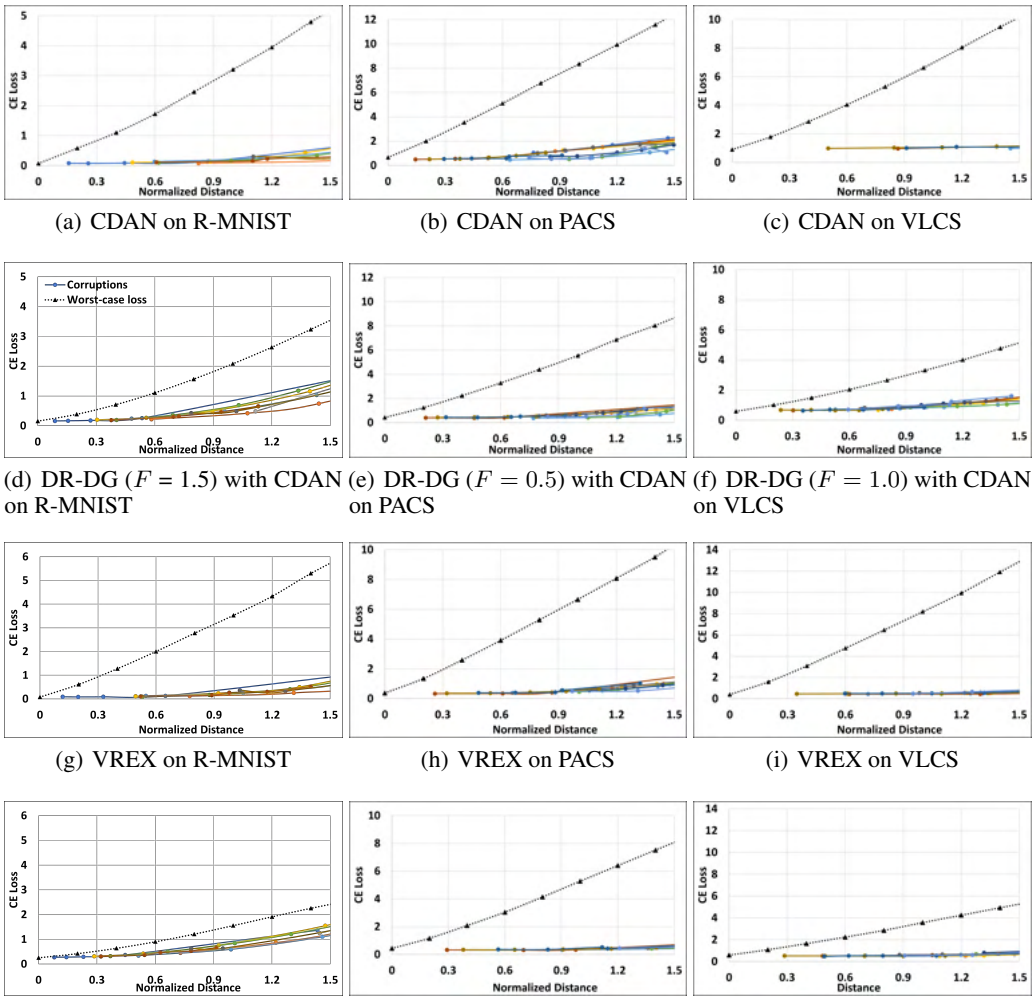
trained by DR-DG. This small gap demonstrates that our representation space-based evaluation is not significantly overestimating the loss of the worst-case distribution compared to the input space evaluation. We thus prefer solving Eq. 2 due to its ease of computation compared to Eq. 1 which gives us the ability to use WC-DG and DR-DG on even large-scale problems.

B.3 Point-wise adversarial robustness versus distributional robustness

Point-wise adversarial robustness deals with worst-case perturbation of a point (usually within an ℓ_p ball). In contrast, distributional robustness,

$$\sup_{P:W_2(P,Q)\leq\rho} \mathbb{E}_{(x,y)\sim P}[\ell(h(g(x)),y)], \tag{6}$$

deals with the worst-case distribution that maximizes the loss of the model on average. Specifically, distributional robustness does not focus on how the model behaves for a specific point from the



(a) CDAN on R-MNIST (b) CDAN on PACS (c) CDAN on VLCS

(d) DR-DG (F = 1.5) with CDAN on R-MNIST (e) DR-DG ($F = 0.5$) with CDAN on PACS (f) DR-DG ($F = 1.0$) with CDAN on VLCS

(g) VREX on R-MNIST (h) VREX on PACS (i) VREX on VLCS

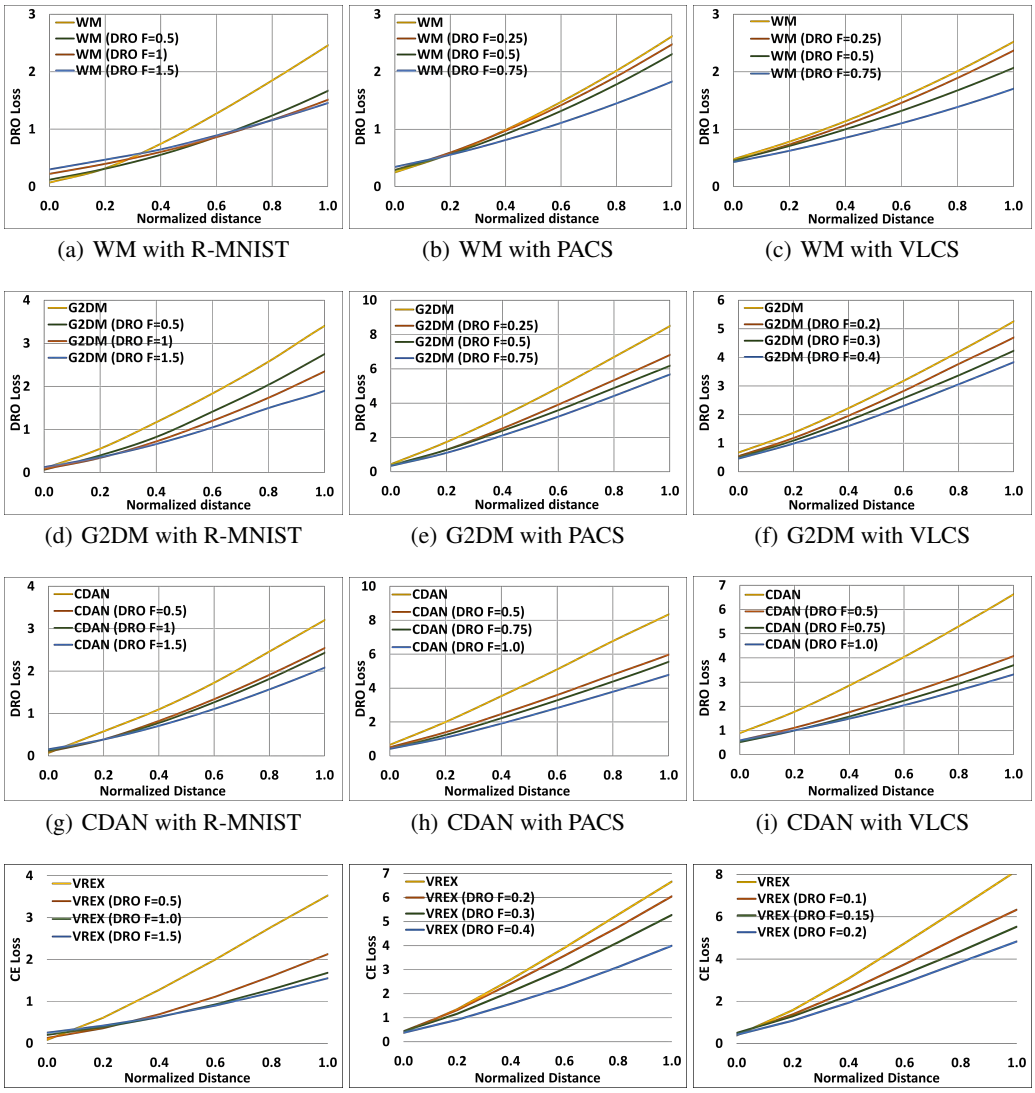
(j) DR-DG (F = 1.5) with VREX on R-MNIST (k) DR-DG ($F = 0.3$) with VREX on PACS (l) DR-DG ($F = 0.2$) with VREX on VLCS

Figure 6: (Best viewed in color.) Worst-case loss of the models trained with CDAN and VREX methods on R-MNIST, PACS, and VLCS. Rows 1 and 3 show models trained with Vanilla DG methods and rows 2 and 4 show models trained with DR-DG using additional losses from CDAN and VREX. The models trained with DR-DG incur smaller worst-case losses compared to their vanilla counterparts and only slightly higher loss on unseen distributions created through common corruptions.

distribution. In this context, the worst-case and average-case point-wise robustness of the models can be evaluated using the following objectives

$$\mathbb{E}_{(x,y)\in Q}[\max_{\delta\in\Delta(x)}[\ell(h(g(x+\delta)),y)]] \text{ and } \mathbb{E}_{(x,y)\in Q}[\mathbb{E}_{\delta\in T(x)}[\ell(h(g(x+\delta)),y)]]$$

where $\Delta(x)$ denotes the permissible perturbations set around a point x, $T(x)$ is the distribution of permissible perturbations and Q is the source distribution. Notice the above problems are different from the problem in Eq. 6, which focuses on generating a worst-case distribution in the Wasserstein ball around the source. The above problems can be approximately solved using a PGD [40] attack and by evaluating the model's performance on known transformations in $T(x)$ but solving for the worst-case distribution as required in Eq. 6 is not possible directly since the problem is infinite-dimensional. Methodology to solve the problem in Eq. 6 is the focus of our work and has been discussed extensively in Sec. 2. Moreover, our work focuses on solving the problem efficiently and computing the loss of the model on the worst-case distribution in the Wasserstein ball around the source distribution in the representation space.



(a) WM with R-MNIST (b) WM with PACS (c) WM with VLCS

(d) G2DM with R-MNIST (e) G2DM with PACS (f) G2DM with VLCS

(g) CDAN with R-MNIST (h) CDAN with PACS (i) CDAN with VLCS

(j) VREX with R-MNIST (k) VREX with PACS (l) VREX with VLCS

Figure 7: (Best viewed in color.) Worst-case loss of models trained with different DG methods using DR-DG proposed in Alg. 2 with three different values of the factor F on R-MNIST, PACS, and VLCS datasets.

Our evaluation algorithm, WC-DG, computes the loss of the worst-case distribution in the Wasserstein ball in the representation space. Since we consider the distribution in the representation space, we are able to solve Eq. 2, without requiring any prohibitive assumptions, which are needed if the worst-case distribution is considered in the input space [56]. Similar to point-wise robustness results [40, 13], models trained with Vanilla DG methods incur high worst-case loss even close to the source distribution. However, this gap does not imply that the worst-case loss is overestimated but shows the existence of distributions that could lead to a much higher loss. Although, by imposing additional assumptions on the target distributions it may be possible to obtain better worst-case loss but it limits the kind of distribution shifts that the model is being evaluated against and is less general than the setting we consider. To improve the worst-case loss of the models, we proposed the DRO-based training approach, DR-DG. The point-wise analog of which are approaches [52, 66, 40] provided new training methods that can make models more robust. Similar to their results we observe that DR-DG lowers the worst-case loss of the models under distribution shift without significantly lowering the empirical performance of the models.

C Additional experimental results

C.1 Variability in the performance of models trained with DG methods on R-MNIST

As discussed in the introduction and Sec. 3, models trained with different DG methods can have high variability in their performance. Surprisingly, this variability happens even on distributions at the same distance from the sources in the representation space. In addition to Fig. 1, Fig. 4 demonstrates this variability on models trained with WM, G2DM[1], CDAN [39], and VREX [33] on the R-MNIST dataset. The variability in the performance of models on unseen distributions lying at the same distance from the sources in the representation space makes it hard to evaluate the generalization performance of DG models using only a few benchmark datasets.

C.2 Detailed worst-case loss results on models trained with DG methods

Here we present detailed results (Rows 1 and 3 of Figs. 5 and 6) of using our evaluation method WC-DG on models trained using WM, G2DM [1], CDAN [39] and VREX[33] on R-MNIST, PACS, and VLCS datasets as presented in Sec. 3. Results in rows 1 and 3 of Figs. 5 and 6 show that models trained with vanilla DG methods incur high worst-case loss even very close to the source. The high worst-case loss implies that the performance of DG methods can vary substantially on different distributions at the same distance and thus good performance on certain benchmark datasets is not enough to guarantee good generalization performance of these methods on unseen domains. Lastly, in addition to the worst-case loss, we also plot loss on different corruption of the data in Figs. 5 and 6. We observe that worst-case loss is much higher than the loss on the corrupted versions of the source test sets which indicates the existence of other realistic unseen distributions that can lower the performance of the DG models.

C.3 Evaluating the worst-case loss of models trained with DR-DG

To improve the worst-case loss of DG models we train the models with DR-DG Alg. 2, which augments the source domain data with examples that incur high loss under the current model. DR-DG training improves the worst-case loss in a Wasserstein ball of radius ρ. The algorithm computes the value of ρ relative to a reference distribution in the representation space (see App. D) and requires a factor, F, as input for it. A large value of F implies a large radius ρ of the Wasserstein ball will be considered for finding the worst-case samples. Similar to point-wise adversarial robustness procedures such as adversarial training [40] or randomized smoothing [13], DR-DG with larger values of ρ can improve the worst-case loss but can lead to a reduction in the performance of the model close to the source distributions. Thus, there exists a trade-off between lowering the worst-case loss at a larger distance versus maintaining the low loss near the source(s). To demonstrate this trade-off we use three different values of F. The results in Figs. 7, 8 and rows 1 and 3 of Figs. 5 and 6 demonstrate that models trained with DR-DG achieve the significantly better worst-case loss in comparison to the models trained with vanilla DG methods with only a minor increase in the loss on the source distribution(s).

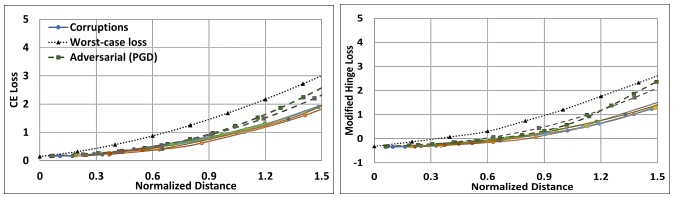


Figure 8: (Best viewed in color.) Improved worst-case loss (with cross-entropy and hinge loss) with models trained with DR-DG using cross-entropy loss using $F = 0.5$ using WM on R-MNIST.

C.4 Comparison of accuracy of models trained with and without DR-DG

Here, we provide a comparison of the accuracy of the models trained with DR-DG and models trained with vanilla DG methods on distributions generated by adding common corruptions to the test set. Fig. 9 shows that the accuracy of DR-DG models doesn't degrade significantly compared to models trained with vanilla methods on these distributions. We emphasize that improving the performance on the distributions of corrupted data is not the primary goal of DR-DG and these results are presented for completeness.

Table 1: DR-DG trained models on two source domains from R-MNIST ($F_1 = 0.5, F_2 = 1, F_3 = 1.5$) and PACS ($F_1 = 0.25, F_2 = 0.5, F_3 = 0.75$), incur small worst-case loss (WC loss). DR-DG also helps reduce the gap between the empirical loss of the models on unseen domains in these datasets and the worst-case loss. This makes the performance of DG models on benchmark datasets more representative of their performance on unseen domains lying at a similar distance.

	30°				45°				Art				Photo			
Method	Accuracy	Loss	WC Loss	Gap	Accuracy	Loss	WC Loss	Gap	Accuracy	Loss	WC Loss	Gap	Accuracy	Loss	WC Loss	Gap
WM	92.11	0.26	1.87	1.61	65.59	1.21	3.76	2.55	73.24	0.82	2.63	1.82	91.55	0.35	2.14	1.79
DR-DG$_{F_1}$	89.61	0.38	0.85	**0.47**	61.12	1.24	2.02	**0.82**	71.48	0.92	2.21	**1.29**	85.93	0.49	1.87	**1.39**
DR-DG$_{F_2}$	88.25	0.52	0.80	**0.28**	54.26	1.36	1.78	**0.44**	69.58	1.01	2.12	**1.11**	87.12	0.52	1.60	**1.08**
DR-DG$_{F_3}$	86.90	0.61	0.87	**0.26**	54.46	1.41	1.77	**0.36**	74.23	0.88	1.53	**0.64**	89.41	0.48	1.27	**0.79**
G2DM	93.86	0.24	4.54	4.29	70.58	1.25	7.21	5.97	75.68	1.16	11.57	10.41	89.64	0.45	11.23	10.78
DR-DG$_{F_1}$	93.24	0.23	3.27	**2.99**	69.23	1.14	5.34	**4.20**	69.14	1.19	9.61	**8.42**	84.43	0.58	9.51	**8.93**
DR-DG$_{F_2}$	90.43	0.31	2.24	**1.93**	63.72	1.24	4.06	**2.82**	67.23	1.32	9.59	**8.23**	89.40	0.41	9.45	**9.05**
DR-DG$_{F_3}$	87.94	0.39	1.48	**1.08**	60.08	1.25	2.84	**1.58**	70.65	1.05	8.26	**7.22**	85.74	0.50	8.29	**7.78**

C.5 Performance of models trained with DR-DG on benchmark datasets

In this section, we evaluate the performance of models trained with DR-DG, Alg. 2, on various target distributions used for comparing the performance of DG methods in previous works. As demonstrated in Table 1, our algorithm improves the worst-case loss of the models for all benchmark distributions. This reduces the gap between the empirical loss of the model on a particular unseen distribution and the worst-case distribution at that distance (from the sources). The reduced gap not only leads to a reduced variability in the performance of the model on target distributions lying at that distance but also makes an evaluation on benchmark datasets more representative of the generalization of the models to unseen distributions. However, gaining robustness to the worst-case distributions may lead to a decrease in the performance of the model on sources as well as on some unseen domains similar to the accuracy vs robustness trade-off observed in point-wise adversarial robustness literature.

D Experimental details

All codes are written in Python using Tensorflow/Keras and were run on Intel Xeon(R) W-2123 CPU with 64 GB of RAM and dual NVIDIA TITAN RTX. Dataset details and model architectures used are described below. We used the Python OT library [20] to measure Wasserstein distances. As described in Sec. 2, we use the type-2 Wasserstein distance in our work. Since the representation space can have an arbitrary scale, we normalize all representation space distances by the distance between the sources and a reference distribution in the representation space. Alg. 2 also relies on the computation of this reference distribution. This reference distribution $P_{S_{adv}}$ consists of points (z', y) generated similarly to the CW-attack [53, 11]: for each z from the source, z' is the closest misclassified point ($h(z') \neq y$). Using this, we report all distances in this paper as the normalized distance $\frac{W_2(P_S,\cdot)}{\rho_{adv}:=W_2(P_S,P_{S_{adv}})}$. The unit distance ρ_{adv} (see Fig. 10) is dependent on the representation space and the classification boundary. As the reference distribution $P_{S_{adv}}$ is the closest distribution whose accuracy is zero (since all points z' are misclassified by construction), the normalized distance of 1 provides a sense of distance from the source where the generalization performance is expected to be low in general (and in particular the accuracy is zero for the reference distribution $P_{S_{adv}}$). We use the Cleverhans [47] implementation of the CW attack [12] in the representation space for computing this distribution.

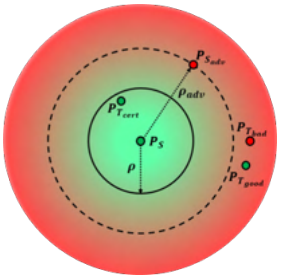


Figure 10: (Best viewed in color.) The space of probability distributions with Wasserstein distance as a metric. P_S and P_T are the sources and the unseen target distributions. The color indicates the performance of the given model on a distribution (Green: good, Red: bad). The worst-case loss (gradually changing color from the center to outside) can be efficiently computed for any distance ρ and is independent of targets. The distribution $P_{S_{adv}}$ is a reference distribution whose distance from the source serves as a unit to normalize distances.

D.1 Dataset description

We have used three popular DG benchmark datasets in our work as described below.

Rotated MNIST (R-MNIST) [24]: This is a variant of the popular MNIST dataset where domains are created by rotating images at different angles. The rotation angles present in the datasets are $\{0, 15, 30, 45, 60, 75\}$. Domains in our dataset comprise 2000 images from each rotation angle. We use 1000 images from the domain for our model training and reserve the other 1000 images for testing. For evaluating the performance of models trained with DG we use the images from the test set (although for unseen domains distinguishing training and test set is not required).

PACS [38]: This dataset contains images for different styles from four domains Art, Cartoons, Photos, and Sketches. It contains 9991 images belonging to 7 different classes. For evaluating the performance of DG methods on source domains we hold out 10% of the data from each source domain. However, for evaluating performance on the unseen domains use all the data from those domains.

VLCS [19]: This dataset contains images from four domains Caltech101, LabelMe, SUN09, VOC2007. It contains 10729 images belonging to 5 different classes. Similar to PACS, we evaluate the performance of DG methods on source domains using a held-out dataset comprising 10% of the data from each source domain and use all the data from unseen domains for evaluating the performance of the models.

D.2 Model description

The representation network for R-MNIST uses a convolutional neural network similar to the one described in Table 7 [25] and we fine-tune a Resnet-50 model for PACS and VLCS. Additionally, for PACS and VLCS we add 2 additional fully connected layers on top of the 512-dimensional output of the Resnet-50 to reduce the size of the output dimension to 128. This is needed since estimating Wasserstein distance in a high dimensional space required a large number of samples. Due to the limited size of these datasets and the even smaller size of the test sets, we reduce the dimension to estimate the Wasserstein distance better. For discriminators used in G2DM and CDAN, we use two fully connected layers and feed in the output of the representation network. For G2DM we do not use the random projection layers in the network and for CDAN we use multi-linear conditioning. Our classifier just comprises a fully connected layer on top of the representation network.

D.3 Corruptions used for creating unseen distributions

To evaluate the performance of models trained with DG methods on unseen domains in relation to their distance from the source domains, we chose to add common corruptions to the test set of the source domain data. The ability to change the severity of the corruption allows us to create multiple unseen domains at different distances from the sources. We emphasize that the use of common corruptions is just an easy and efficient way of generating multiple unseen domains and only covers a small subset of all possible unseen domains. Moreover, our task here is not to show that the performance of models trained with different DG methods deteriorates on these corrupted domains but it is rather to highlight the high variability in their performance even when distributions lie at the same distance. For R-MNIST we use the corruptions from the MNIST-C dataset [46] (`shot noise, impulse noise, glass blur, shear, scale, translate, fog, spatter, elastic transform`) and for PACS and VLCS we use corruptions from the Imagenet-C dataset [28] (`Gaussian noise, shot noise, impulse noise, speckle noise, Gaussian blur, defocus blur, zoom blur, spatter, contrast, brightness, saturate, elastic transform`).

D.4 PGD adversarial attacks

We consider adversarial attacks using PGD [40] to demonstrate the existence of distributions beyond common corruptions which can degrade the performance of DG methods. We consider two variants of this attack. The first is the usual PGD attack where the adversarial examples are crafted in the input space by maximizing the loss of the perturbed examples (denoted by the grey dashed line with square markers in Fig. 3). The distance between the adversarial example and the clean example is measured in the input space in this case. The second variant is the PGD attack created in the representation

space. In this attack, the distortion between the clean and adversarial examples is measured in the representation space and the perturbation is applied to the representation of the clean examples, i.e., $\max_{z'} \ell(h(z'), y)$ s.t. $\|z' - z\|_2 \leq \epsilon_g$ where z' is the adversarial example in the representation space for the point $(z = g(x), y)$ and ϵ_g is the bound on the distortion in the representation space. The result of this attack is shown as the green dashed line in Fig. 3).

D.5 Implementation details of WC-DG

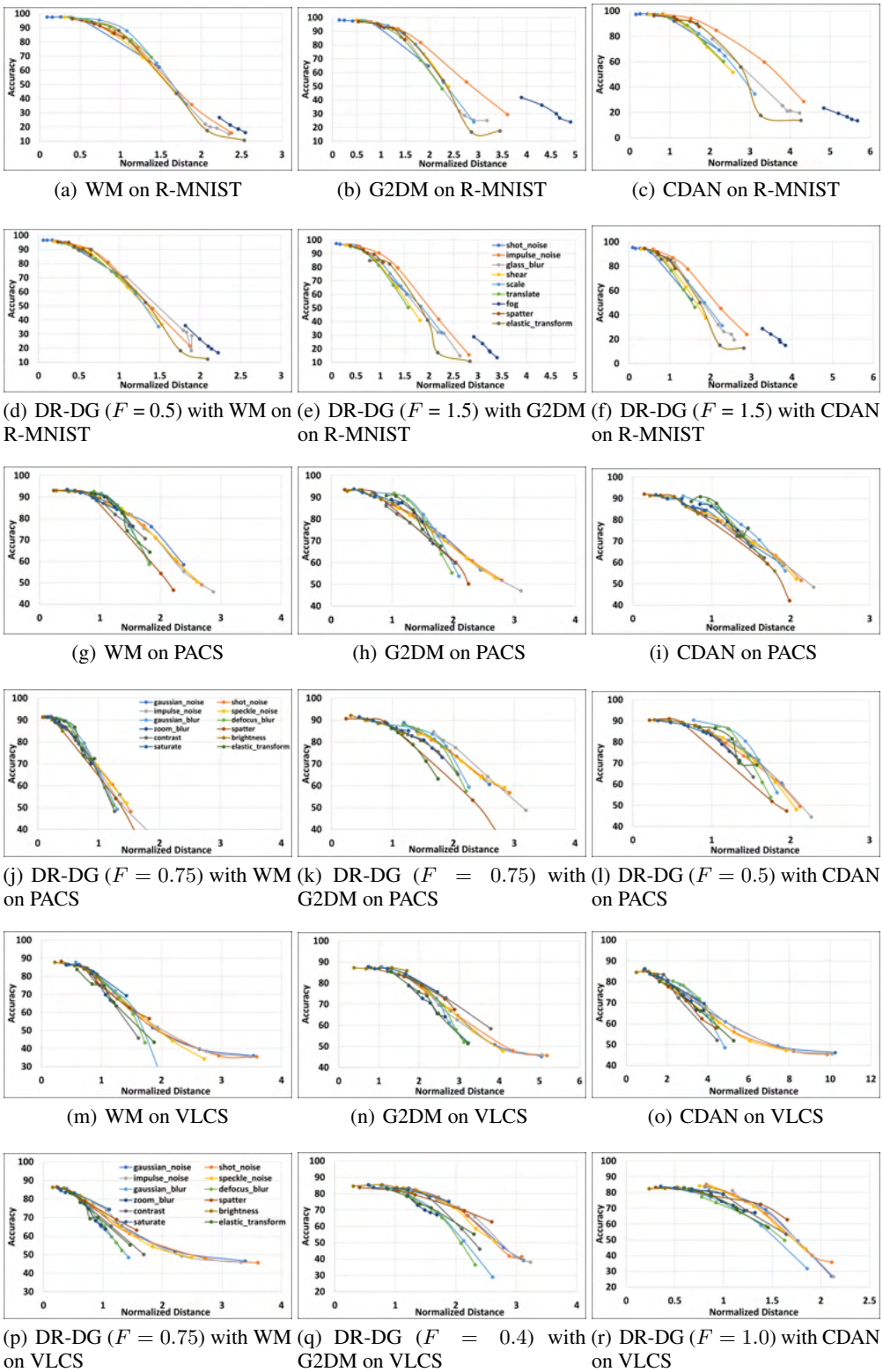
Here we briefly describe some of the implementation details for our evaluation and training algorithms. WC-DG requires computing distributions that have an average distortion of ρ^2. Solving the Eq. 2 requires solving the inner maximization for every point. We solve the problem batch-wise and keep track of the perturbation applied to the points. Since the network parameters are fixed during evaluation, using the previously saved perturbation as the starting point helps the algorithm converge faster with the minimal tuning of the hyperparameters such as the learning rates in Alg. 1.

E Related Work

Domain generalization and domain adaptation: The key problem addressed by the works in these areas is to improve the performance of the models when training and test distributions are different. Several analytical works [5, 4, 41, 55, 43, 70, 31, 9] have demonstrated that the performance of the models remains high under distribution shifts if the training (source) and test (target) distributions are close under certain divergence measures. Distributional divergence measure studied in previous works includes the Wasserstein distance [56, 35, 34, 23, 16], maximum mean discrepancy [57], f-divergence [6, 64], and $\mathcal{H}$-divergence [5, 1]. Since generalization to arbitrary domains is not possible, previous works make additional assumptions on the unseen domains such as [8] assuming that source and target distributions are derived from the same hyper-distribution, [1, 33] assuming that target distributions belong to the convex hull of the source distributions(s), [35] considers shifts generated by different parameterized transformations, [64] considers shifts with bounded mean and variance difference between the two distributions. Another line of work considers learning a representation space by minimizing different divergence measures between the source distributions [1, 67, 22, 71, 49, 25]. A different line of research in DG learns a representation space that disentangles [3, 68, 17, 45] domain-specific features from the domain invariant features such that predictors learned on top of these invariant features also become domain invariant. Yet another popular approach for DG is to use data augmentation [29, 62, 32, 10, 58] to make the models invariant to common variations of the data from the source.

Certified robustness: Point-wise evaluation of the performance of the classifier has been extensively studied in the area of certified adversarial robustness [59, 40, 36, 37, 65, 50, 52, 66, 56, 42], where the evaluation outputs a radius around a test point within which classifier predictions remain constant. Our work is different from these works since we are concerned with the distributional robustness of DG models, i.e., a lower bound on the performance of the models at a particular distance in the representation space which quantifies the performance on an unseen distribution rather than certifying instance-wise performance. Recently, certified robustness has gained attention in the context of certifying the performance of the classifier (in a distributional sense) on bounded distribution shifts [35, 64, 53].

Distributionally robust optimization: DRO has been extensively used in machine learning to quantify the performance of a model on distributions belonging to different uncertainty sets [6, 44, 18, 69, 23, 7]. Previous works have considered the uncertainty sets to be the Wasserstein balls [56, 23, 54, 14, 34, 9, 61, 49], f-divergence balls [6, 18, 26, 30, 64] around the source distribution as well as shifts over different groups [51]. Our evaluation methodology considers the uncertainty set to be Wasserstein balls in the representation space.



(a) WM on R-MNIST (b) G2DM on R-MNIST (c) CDAN on R-MNIST

(d) DR-DG (F = 0.5) with WM on R-MNIST (e) DR-DG (F = 1.5) with G2DM on R-MNIST (f) DR-DG (F = 1.5) with CDAN on R-MNIST

(g) WM on PACS (h) G2DM on PACS (i) CDAN on PACS

(j) DR-DG ($F = 0.75$) with WM on PACS (k) DR-DG ($F = 0.75$) with G2DM on PACS (l) DR-DG ($F = 0.5$) with CDAN on PACS

(m) WM on VLCS (n) G2DM on VLCS (o) CDAN on VLCS

(p) DR-DG ($F = 0.75$) with WM on VLCS (q) DR-DG ($F = 0.4$) with G2DM on VLCS (r) DR-DG ($F = 1.0$) with CDAN on VLCS

Figure 9: (Best viewed in color.) Comparison of accuracy of the models on unseen distributions created by adding common corruptions to the source test set trained with Vanilla DG methods (rows 1, 3, and 5) and DR-DG (rows 2, 4, and 6) on R-MNIST (rows 1 and 2), PACS (rows 3 and 4), and VLCS (rows 5 and 6).