RLSBENCH: A Large-Scale Empirical Study of Domain Adaptation Under Relaxed Label Shift

Anonymous Author(s)
Affiliation
Address
email

Abstract

Despite the emergence of principled methods for domain adaptation under label shift (where only the class balance changes), the sensitivity of these methods to natural-seeming covariate shifts remains precariously underexplored. Meanwhile, popular deep domain adaptation heuristics, despite showing promise on benchmark datasets, tend to falter when faced with shifts in the class balance. Moreover, it's difficult to assess the state of the field owing to inconsistencies among relevant papers in evaluation criteria, datasets, and baselines. In this paper, we introduce RLSBENCH, a large-scale benchmark for such *relaxed label shift* settings, consisting of 11 vision datasets spanning >200 distribution shift pairs with different class proportions. We evaluate 12 popular domain adaptation methods, demonstrating a more widespread susceptibility to failure under extreme shifts in the class proportions than was previously known. We develop an effective meta-algorithm, compatible with most deep domain adaptation heuristics, that consists of the following two steps: (i) *pseudo-balance* the data at each epoch; and (ii) adjust the final classifier with (an estimate of) target label distribution. Furthermore, we discover that batch-norm adaption of a model trained on source with aforementioned corrections offers a strong baseline, largely missing from prior comparisons. We hope that these findings and the availability of RLSBENCH will encourage researchers to include rigorously evaluate proposed methods in relaxed label shift settings.

1 Introduction

Real-world deployments of machine learning models are typically characterized by distribution shift, where data encountered in production exhibits statistical differences from the available training data [52, 72, 34]. Because continually labeling data can be prohibitively expensive, researchers have focused on the unsupervised domain adaptation (DA) setting, where only labeled data sampled from the *source* distribution and unlabeled from the *target* distribution are available for training.

Absent further assumptions, the DA problem is well known to be underspecified [6] and thus no method is universally applicable. Researchers have responded to these challenges in several ways. One approach is to investigate additional assumptions that render the problem well-posed. Popular examples include covariate shift and label shift, for which identification strategies and principled methods exist whenever the source and target distributions have overlapping support [63, 62, 25]. Under label shift in particular, recent research has produced effective methods that are applicable in deep learning regimes and yield both consistent estimates of the target label marginal and principled ways to update the resulting classifier [38, 1, 3, 23]. However, these assumptions are typically, to some degree, violated in practice. Even for archetypal cases like shift in disease prevalence [38], the label shift assumption can be violated. For example, over the course of the COVID-19 epidemic, changes in disease positivity have been coupled with shifts in the age distribution of the infected and subtle mutations of the virus itself.

Submitted to the Workshop on Distribution Shifts, 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

A complementary line of research focuses on constructing benchmark datasets for evaluating methods, in the hopes of finding heuristics that, for the kinds of problems that arise in practice, tend to incorporate the unlabeled target data profitably. Examples of such benchmarks include OfficeHome [75], Domainnet [50]), WILDS [59]. However, most academic benchmarks exhibit little or no shift in the label distribution $p(y)$. Consequently, benchmark-driven research has produced a variety of heuristic methods [21, 64, 76, 37] that despite yielding gains in benchmark performance tend to break when $p(y)$ shifts. While this has previously been shown for domain-adversarial methods [80, 90], we show that this problem is more widespread than previously known. Several recent papers attempt to address shift in label distribution compounded by natural variations in $p(x|y)$ [70, 69, 51]. However, the experimental evaluations are hard to compare across papers owing to discrepancies in how shifts in $p(y)$ are simulated and the choice of evaluation metrics. Moreover, many methods violate the unsupervised contract by peeking at target validation performance during model selection and hyperparameter tuning.

In this paper, we develop a test bed of *relaxed label shift* settings, where $p(y)$ can shift arbitrarily and the class conditionals $p(x|y)$ can shift in seemingly natural ways (following the popular DA benchmarks). Using RLSBENCH, we evaluate a collection of popular DA methods based on domain-invariant representation learning, self-training, and test-time adaptation methods across 11 multi-domain datasets. The different domains in each dataset present a different shift in $p(x|y)$. Since these datasets exhibit minor to no shift in label marginal, we simulate shift in target label marginal via stratified sampling with varying severity. Overall, we obtain 220 different source and target distribution shift pairs and train $> 10k$ models in our testbed.

First, we observe that while popular DA methods often improve over a source only classifier absent shift in target label distribution, their performance tends to degrade, dropping below source-only classifiers under severe shifts in target label marginal. Next, we show that in these relaxed label shift settings, the performance of DA methods tends to improve when paired with a meta-algorithm with two simple corrections: (i) re-sampling the data to balance the source and pseudo-balance the target; (ii) re-weighting the final classifier using an estimate of target label marginal. Overall, we observe that popular DA methods (e.g. FixMatch and BN-adapt) when combined with corrections (i) and (ii) often improve over methods specifically proposed for relaxed label shift (e.g., IW-CDANN and SENTRY).

2 RLSBENCH: A Benchmark for Relaxed Label Shift

In the traditional label shift setting, one asssumes that $p(x|y)$ does not change but that $p(y)$ can. This paper focuses on the *relaxed label shift* setting. In particular, we assume that the label distribution can shift from source to target arbitrarily but that $p(x|y)$ varies between source and target in some comparatively subtle way. We keep this definition mathematically imprecise as we lack a rigorous characterization of the sense in which those shifts addressed in popular DA benchmarks are natural. Here, given access to labeled source data and unlabeled target data, our goals are: (i) estimate the target label marginal $p_t(y)$; and (ii) train a classifier f to maximize the performance on target domain.

We now introduce RLSBENCH, a suite of datasets and domain adaptation algorithms that are at the core of our benchmark study. Motivated by correction methods for the (stricter) label shift setting [58, 38] and learning under imbalanced datasets [77, 11], we also present simple corrections that we incorporate in our benchmark to tackle a shift in target marginal.

Datasets RLSBENCH builds on eleven open-source multi-domain datasets for image classification spanning applications in object classification, satellite imagery and medicine. Across our datasets, we obtain a total of 44 different source and target pairs. We relegate details about the datasets in App. F.

Simulating a shift in target marginal The above datasets present minor to no shift in label marginal. Hence, we simulate such a shift by altering the target label marginal and keeping the source target distribution fixed (to the original source label distribution). Note that, unlike some previous studies, we do not alter the source label marginal because in practice, we may have an option to carefully curate the training distribution but might have little to no control over the test data. For each target dataset, we have the true labels which allow us to vary the target label distribution. In particular, we sample the target label marginal from a Dirichlet distribution with a parameter $\alpha \in \{0.5, 1, 3.0, 10\}$ multiplier to the original target marginal. Specifically, $p_t(y) \sim \mathrm{Dir}(\beta)$ with $\beta_y = \alpha \cdot p_{t,0}(y) \cdot k$ where $p_{t,0}(y)$ is the original target label marginal and k is the number of classes. The Dirichlet parameter α controls the severity of shift in target label marginal. Intuitively, as α decreases, the severity in shift increases. For completeness, we also include the target dataset with the original target label marginal (we denote this as NONE in the set of Dirichlet parameters, i.e., the limiting distribution as $\alpha \to \infty$). After simulating shift in the target label marginal, we obtain 220 pairs of different source and target datasets.

Domain Adaptation Methods With the current version of RLSBENCH, we implement the following algorithms (a more detailed description of each method is included in App. H): (i) *Source only:* As a baseline, we include model trained with empirical risk minimization with cross-entropy loss on the source domain. We also include adversarial robust models; (ii) *Domain alignment methods:* These methods employ domain-adversarial training aimed to learn invariant representations across different domains [21, 89, 70]; In particular, we include: **DANN** [21], **CDAN** [42], Importance-reweighted DANN (i.e., **IWDAN**) and CDAN (i.e., **IWCDAN**) [69]; (iii) *Self-training methods:* These methods *pseudo-label* unlabeled examples with the model's own predictions and then train on them as if they were labeled examples [36, 81, 7]. We include the following algorithms: **FixMatch** [64], **Noisy Student** [81], **SENTRY** [51]; (iv) *Test-time adaptation methods:* These methods take a source trained model and adapt few parameters (e.g. batch norm parameters, batch norm statistics) on the unlabeled target data. We include the following methods: **CORAL** [66], **BN-adapt** [37, 61], **TENT** [76].

2.1 Meta Algorithm to handle shifts in target class proportions

Here we discuss two simple general-purpose corrections that we implement in our framework. First, note that, as the severity of shift in the target label marginal increases, the performance of DA methods can falter as the training is done over source and target datasets with different class proportions. Indeed, failure of domain adversarial training methods (one category of deep domain adaptation methods) has been theoretically and empirically shown in the literature [80, 90]. In our experiments, we show that a failure due to a shift in label distribution is not limited to domain adversarial training methods, but is common with all the popular DA methods (Sec. 3).

Re-sampling To handle label imbalance in standard supervised learning, re-sampling the data to balance the class marginal is a known successful strategy [13, 9, 11]. In relaxed label shift, we seek to handle the imbalance in the target data (with respect to the source label marginal), where we do not have access to true labels. We adopt an alternative strategy of leveraging pseudolabels for target data to perform pseudo class-balanced re-sampling [91, 77]. For relaxed label shift problems, Prabhu et al. [51] employed this technique with their SENTRY objective. However, they did not explore re-sampling based correction for existing DA techniques. Since this technique can be used in conjunction with any DA methods, we employ this re-sampling technique with existing DA methods and find that re-sampling benefits all DA methods, often improving over SENTRY in our testbed (Sec. 3).

Re-weighting With re-sampling, we can hope to train the classifier f on a mixture of balanced source and balanced target datasets in an ideal case. However, this still leaves open the problem of adapting the classifier f to the original target label distribution which is not available. If we can estimate the target label marginal, we can adapt the classifier f with a simple re-weighting correction [38, 1]. To estimate the target label marginal, we turn to techniques developed under the stricter label shift assumption (recall, the setting where $p(x|y)$ remains domain invariant). This also allows us to empirically evaluate efficacy of label shift estimation methods when we begin violating the conditions required for consistency of these techniques. We provide precise details about label shift estimation methods in App. G. Since these methods leverage off-the-shelf classifiers, classifiers obtained with any deep DA methods can be used in conjunction with these estimation methods.

Summary Overall, Algorithm 1 discusses how to incorporate the re-sampling and re-weighting correction with existing DA techniques. Algorithm $\mathcal{A}$ can be any DA methods and we can use any of the label shift estimation methods to estimate the target label marginal in Step 7. In an ideal scenario, we expect DA methods to adapt classifier f to $p(x|y)$ shift and our meta-algorithm to adapt f to shift in $p(y)$. We emphasize that in our work, we *do not* claim to propose these corrections. But, to the best of our knowledge, our work is the first to combine these two corrections together in relaxed label shift scenarios and perform extensive experiments across diverse datasets.

3 Main Results

For a fair comparison, we re-implemented all the algorithms with consistent design choices. For our main experiments, we perform model selection with source validation performance. Other implementation choices are described in App. E. We present aggregated results in Table 1. In Table 2, we include results with Re-Sampling (RS) and Re-Weighting (RW) corrections. Results with individual methods and shifts in App. N. Based on running the entire suite, we distill our findings into the following takeaways:

Popular deep DA methods fail without any correction. While DA methods typically improve over a source only classifier for cases when shift in target label marginal is absent or low, performance of these methods (except Noisy Student) drops below the performance of a source only classifier when the shift in target label marginal is severe (i.e., when $\alpha = 0.5$ in Table 1). With RS and RW correction, we can avoid this failure mode (and rather observe improvements in Table 2).

Re-sampling to pseudobalance target often helps all DA methods. When the shift in target label marginal is absent or small (i.e., $\alpha \in \{\text{NONE}, 10.0\}$ in Table 2), we observe no (significant) differences in performance with re-sampling. However, as the shift severity in increases (i.e., $\alpha \in \{3.0, 1.0, 0.5\}$ in Table 2), we observe that re-sampling typically improves all DA methods in our testbed.

Effect of re-weighting the classifier depends on the nature of shift. We observe that in certain scenarios of real-world shift in $p(x|y)$ (e.g., subpopulation shift in BREEDs datasets, camelyon shifts, and replication study in CIFAR-10), re-weighting the classifier with a target label marginal estimate helps in cases when there is shift in target label marginal and does no harm in cases without any shift (ref. to Table 2 for aggregated results and ref. to App. N for individual results). However, in other datasets (e.g., domainnet or officehome where shift is going from real world images to sketches/art), we obtain mixed results. When the shift in target label marginal is absent or low, re-weighting with target label marginal estimate can slightly hurt (i.e., $\alpha \in \{\text{NONE}, 10.0\}$ in Table 2). On the other hand, when the target label marginal shift is large, re-weighting with an estimate of target label marginal can significantly improve performance of all methods (i.e., $\alpha \in \{3.0, 1.0, 0.5\}$ in Table 2). Note that in all the cases, RW with true target marginal consistently helps (ref. to individual results in App. N).

Improvement over source only classifier with DA methods but no method consistently performs the best. First, we observe that our source only numbers are better than previously published results. Similar to previous studies [26], this can be attributed to improved design choices (e.g. data augmentation, hyperparameters). While no method consistently does the best across datasets, FixMatch with RS and RW provides the highest overall improvement over a source only model.

Batch Norm adaptation is a simple and strong baseline. For models with batch norm parameters, BN-adapt with RS and RW is a computationally efficient and strong baseline. We observe that while the performance of BN-adapt can drop substantially when target label marginal shifts (i.e., $\alpha \in \{1.0, 0.5\}$ in Table 2), RS and RW correction improves the performance often improving BN-adapt over all other DA methods when the shift in target marginal is extreme (i.e., $\alpha = 0.5$ in Table 2).

Early stopping criterion matters. We observe a consistent $\approx 2\%$ accuracy difference with all methods, highlighting the importance of better early stopping criteria (oracle results in App. L).

Deep domain adaptation methods improve label marginal estimation. Recall that we experiment with target marginal estimation methods that leverage off-the-shelf classifiers to obtain an estimate. We observe that estimation methods leveraging DA methods tend to perform better than using source only classifiers (RLLS in Table 3 and others in App. M). As one might expect, better estimation yields greater improvements when applying RW correction, favoring DA methods over the source-only classifier (Table 2). Moreover, we observe a trade-off in the performance of the baseline estimator (i.e. binning target pseudolabels) and RLLS (or MLLS) with severity of target marginal shift. When the shift in target label marginal is low (i.e. $\alpha \in \{\text{NONE}, 10.0, 3.0\}$), baseline estimate performs better than RLLS whereas as the shift gets severe (i.e. $\alpha \in \{1.0, 0.5\}$) RLLS improves over baseline.

Comparison with other methods proposed for relaxed label shift. We note that, with consistent experimental design across different methods, existing DA methods with RS and RW correction can often improve over previous methods aimed to tackle relaxed label shift (i.e., IW-CDAN, IW-DAN and SENTRY). While the importance weighting correction (i.e., IW-CDAN and IW-DAN) improves over CDANN and DANN respectively, RS and RW corrections outweight those improvements (Table 1 and Table 2). Similarly, except on Visda dataset, we observe that FixMatch even without RS and RW correction tends to do better than SENTRY. On Visda dataset, SENTRY significantly improves over other DA methods (Table 1). However, with RS and RW correction, we observe that FixMatch improves over SENTRY even on Visda (Table 2). We discuss SENTRY results more in App. J.

4 Conclusion

Our work is the first large-scale study investigating methods under the relaxed label shift scenario. Relative to works operating strictly under the label shift assumption, RLSBENCH provides an opportunity for sensitivity analysis, allowing researchers to measure the robustness of their methods under various sorts of perturbations to the class-conditional distributions. Relative to the benchmark-driven deep domain adaptation literature, our work provides a comprehensive and standardized suite for evaluating under shifts in label distributions, bringing these benchmarks one step closer to exhibiting the sort of diversity that we should expect to encounter when deploying models in the wild.

Reproducibility Statement

Our code with all the results will be released on GitHub with the camera ready submission. We implement our LSBENCH library in PyTorch [48] and provide an infrastructure to run all the experiments to generate corresponding results. We have stored all models and logged all hyperparameters and seeds to facilitate reproducibility. In our appendices, we provide additional details on datasets and experiments. In App. F, we describe dataset information and in App. I, we describe hyperparameter details.

References

[1] A. Alexandari, A. Kundaje, and A. Shrikumar. Adapting to label shift with bias-corrected calibration. In *International Conference on Machine Learning (ICML)*, 2021.

[2] J. An, L. Ying, and Y. Zhu. Why resampling outperforms reweighting for correcting sampling bias with stochastic gradients. *arXiv preprint arXiv:2009.13447*, 2020.

[3] K. Azizzadenesheli, A. Liu, F. Yang, and A. Anandkumar. Regularized learning for domain adaptation under label shifts. In *International Conference on Learning Representations (ICLR)*, 2019.

[4] P. Bandi, O. Geessink, Q. Manson, M. Van Dijk, M. Balkenhol, M. Hermsen, B. E. Bejnordi, B. Lee, K. Paeng, A. Zhong, et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 2018.

[5] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2), 2010.

[6] S. Ben-David, T. Lu, T. Luu, and D. Pál. Impossibility Theorems for Domain Adaptation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2010.

[7] D. Berthelot, R. Roelofs, K. Sohn, N. Carlini, and A. Kurakin. Adamatch: A unified approach to semi-supervised learning and domain adaptation. *arXiv preprint arXiv:2106.04732*, 2021.

[8] G. Blanchard, G. Lee, and C. Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

[9] M. Buda, A. Maki, and M. A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural networks*, 106:249–259, 2018.

[10] J. Byrd and Z. C. Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning (ICML)*, 2019.

[11] Z. Cao, K. You, M. Long, J. Wang, and Q. Yang. Learning to transfer examples for partial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2985–2994, 2019.

[12] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33:9912–9924, 2020.

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[15] G. Christie, N. Fendley, J. Wilson, and R. Mukherjee. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[16] C. Cortes and M. Mohri. Domain adaptation and sample bias correction theory and algorithm for regression. *Theoretical Computer Science*, 519, 2014.

[17] C. Cortes, Y. Mansour, and M. Mohri. Learning Bounds for Importance Weighting. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[18] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[19] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[20] J. Djolonga, J. Yung, M. Tschannen, R. Romijnders, L. Beyer, A. Kolesnikov, J. Puigcerver, M. Minderer, A. D'Amour, D. Moldovan, et al. On robustness and transferability of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16458–16468, 2021.

[21] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 2016.

[22] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[23] S. Garg, Y. Wu, S. Balakrishnan, and Z. Lipton. A unified view of label shift estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[24] S. Gidaris, P. Singh, and N. Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[25] A. Gretton, A. J. Smola, J. Huang, M. Schmittfull, K. M. Borgwardt, and B. Schölkopf. Covariate Shift by Kernel Mean Matching. *Journal of Machine Learning Research (JMLR)*, 2009.

[26] I. Gulrajani and D. Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[28] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[29] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[30] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8349, 2021.

[31] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[32] B. Y. Idrissi, M. Arjovsky, M. Pezeshki, and D. Lopez-Paz. Simple data balancing achieves competitive worst-group-accuracy. In *Conference on Causal Learning and Reasoning*, pages 336–351. PMLR, 2022.

[33] J. Jiang, Y. Shu, J. Wang, and M. Long. Transferability in deep learning: A survey, 2022.

[34] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, T. Lee, E. David, I. Stavness, W. Guo, B. A. Earnshaw, I. S. Haque, S. Beery, J. Leskovec, A. Kundaje, E. Pierson, S. Levine, C. Finn, and P. Liang. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning (ICML)*, 2021.

[35] A. Krizhevsky and G. Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, Citeseer, 2009.

[36] D.-H. Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

[37] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

[38] Z. C. Lipton, Y.-X. Wang, and A. Smola. Detecting and Correcting for Label Shift with Black Box Predictors. In *International Conference on Machine Learning (ICML)*, 2018.

[39] X. Liu, Z. Guo, S. Li, F. Xing, J. You, C.-C. J. Kuo, G. El Fakhri, and J. Woo. Adversarial Unsupervised Domain Adaptation with Conditional and Label Shift: Infer, Align and Iterate. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10347–10356, Montreal, QC, Canada, Oct. 2021. IEEE. ISBN 978-1-66542-812-5. doi: 10.1109/ICCV48922.2021.01020. URL `https://ieeexplore.ieee.org/document/9710205/`.

[40] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

[41] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*. PMLR, 2017.

[42] M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. *Advances in neural information processing systems*, 31, 2018.

[43] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[44] J. Manders, T. van Laarhoven, and E. Marchiori. Adversarial Alignment of Class Prediction Uncertainties for Domain Adaptation, Jan. 2019. URL `http://arxiv.org/abs/1804.04448`. Number: arXiv:1804.04448 arXiv:1804.04448 [cs, stat].

[45] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. *arXiv preprint arXiv:0902.3430*, 2009.

[46] J. P. Miller, R. Taori, A. Raghunathan, S. Sagawa, P. W. Koh, V. Shankar, P. Liang, Y. Carmon, and L. Schmidt. Accuracy on the line: on the strong correlation between out-of-distribution and in-distribution generalization. In *International Conference on Machine Learning*. PMLR, 2021.

[47] K. Muandet, D. Balduzzi, and B. Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013.

[48] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

[49] X. Peng, B. Usman, K. Saito, N. Kaushik, J. Hoffman, and K. Saenko. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. *arXiv preprint arXiv:1806.09755*, 2018.

[50] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

[51] V. Prabhu, S. Khare, D. Kartik, and J. Hoffman. Sentry: Selective entropy optimization via committee consistency for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8558–8567, 2021.

[52] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset shift in machine learning*. Mit Press, 2008.

[53] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[54] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

[55] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[56] E. Rosenfeld, P. Ravikumar, and A. Risteski. Domain-adjusted regression or: Erm may already learn features sufficient for out-of-distribution generalization. *arXiv preprint arXiv:2202.06856*, 2022.

[57] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[58] M. Saerens, P. Latinne, and C. Decaestecker. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. *Neural Computation*, 2002.

[59] S. Sagawa, P. W. Koh, T. Lee, I. Gao, S. M. Xie, K. Shen, A. Kumar, W. Hu, M. Yasunaga, H. Marklund, S. Beery, E. David, I. Stavness, W. Guo, J. Leskovec, K. Saenko, T. Hashimoto, S. Levine, C. Finn, and P. Liang. Extending the wilds benchmark for unsupervised adaptation. In *NeurIPS Workshop on Distribution Shifts*, 2021.

[60] S. Santurkar, D. Tsipras, and A. Madry. Breeds: Benchmarks for subpopulation shift. In *International Conference on Learning Representations (ICLR)*, 2021.

[61] S. Schneider, E. Rusak, L. Eck, O. Bringmann, W. Brendel, and M. Bethge. Improving robustness against common corruptions by covariate shift adaptation. *arXiv preprint arXiv:2006.16971*, 2020.

[62] B. Schölkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij. On Causal and Anticausal Learning. In *International Conference on Machine Learning (ICML)*, 2012.

[63] H. Shimodaira. Improving Predictive Inference Under Covariate Shift by Weighting the Log-Likelihood Function. *Journal of Statistical Planning and Inference*, 2000.

[64] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33, 2020.

[65] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*. Springer, 2016.

[66] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

[67] B. Sun, J. Feng, and K. Saenko. Correlation alignment for unsupervised domain adaptation. In *Domain Adaptation in Computer Vision Applications*. Springer, 2017.

[68] R. Tachet, H. Zhao, Y.-X. Wang, and G. Gordon. Domain Adaptation with Conditional Distribution Matching and Generalized Label Shift. *arXiv:2003.04475 [cs, stat]*, Dec. 2020. URL `http://arxiv.org/abs/2003.04475`. arXiv: 2003.04475.

[69] R. Tachet des Combes, H. Zhao, Y.-X. Wang, and G. J. Gordon. Domain adaptation with conditional distribution matching and generalized label shift. *Advances in Neural Information Processing Systems*, 33, 2020.

[70] S. Tan, X. Peng, and K. Saenko. Class-imbalanced domain adaptation: An empirical odyssey. In *European Conference on Computer Vision*, pages 585–602. Springer, 2020.

[71] R. Taori, A. Dave, V. Shankar, N. Carlini, B. Recht, and L. Schmidt. Measuring robustness to natural distribution shifts in image classification. *Advances in Neural Information Processing Systems*, 33:18583–18599, 2020.

[72] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[73] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1958–1970, 2008.

[74] V. N. Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

[75] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[76] D. Wang, E. Shelhamer, S. Liu, B. Olshausen, and T. Darrell. Tent: Fully test-time adaptation by entropy minimization. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=uXl3bZLkr3c`.

[77] C. Wei, K. Sohn, C. Mellina, A. Yuille, and F. Yang. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10857–10866, 2021.

[78] F. Wenzel, A. Dittadi, P. V. Gehler, C.-J. Simon-Gabriel, M. Horn, D. Zietlow, D. Kernert, C. Russell, T. Brox, B. Schiele, et al. Assaying out-of-distribution generalization in transfer learning. *arXiv preprint arXiv:2207.09239*, 2022.

[79] O. Wiles, S. Gowal, F. Stimberg, S. Alvise-Rebuffi, I. Ktena, T. Cemgil, et al. A fine-grained analysis on distribution shift. *arXiv preprint arXiv:2110.11328*, 2021.

[80] Y. Wu, E. Winston, D. Kaushik, and Z. Lipton. Domain adaptation with asymmetrically-relaxed distribution alignment. In *International Conference on Machine Learning (ICML)*, 2019.

[81] X. Xie, J. Chen, Y. Li, L. Shen, K. Ma, and Y. Zheng. Self-supervised cyclegan for object-preserving image-to-image domain adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 498–513. Springer, 2020.

[82] D. Xu, Y. Ye, and C. Ruan. Understanding the role of importance weighting for deep learning. *arXiv preprint arXiv:2103.15209*, 2021.

[83] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2272–2281, 2017.

[84] B. Zadrozny. Learning and Evaluating Classifiers Under Sample Selection Bias. In *International Conference on Machine Learning (ICML)*, 2004.

[85] J. Zhang, A. Menon, A. Veit, S. Bhojanapalli, S. Kumar, and S. Sra. Coping with label shift via distributionally robust optimisation. In *International Conference on Learning Representations (ICLR)*, 2021.

[86] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang. Domain Adaptation Under Target and Conditional Shift. In *International Conference on Machine Learning (ICML)*, 2013.

[87] R. Zhang. Making convolutional networks shift-invariant again. In *ICML*, 2019.

[88] W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[89] Y. Zhang, T. Liu, M. Long, and M. Jordan. Bridging theory and algorithm for domain adaptation. In *International Conference on Machine Learning*. PMLR, 2019.

[90] H. Zhao, R. T. Des Combes, K. Zhang, and G. Gordon. On learning invariant representations for domain adaptation. In *International Conference on Machine Learning*, pages 7523–7532. PMLR, 2019.

[91] Y. Zou, Z. Yu, B. Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018.

Appendix

A Preliminaries and Prior Work

We first setup the notation and formally define the problem setup. Let $\mathcal{X}$ be the input space and $\mathcal{Y} = \{1, 2, \ldots, k\}$ the output space. Let $\mathrm{P}_s, \mathrm{P}_t : \mathcal{X} \times \mathcal{Y} \to [0, 1]$ be the source and target distributions and let p_s and p_t denote the corresponding probability density (or mass) functions. Unlike the standard supervised setting, in unsupervised DA, we possess labeled source data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ and unlabeled target data $\{x_{n+1}, x_{n+2}, \ldots, x_{n+m}\}$. With $f : \mathcal{X} \to \Delta^{k-1}$, we denote a predictor function which predicts $\widehat{y} = \arg\max_y f_y(x)$ on an input x. For a vector v, we use v_y to access the element at index y.

In the traditional label shift setting, one asssumes that $p(x|y)$ does not change but that $p(y)$ can. Under label shift, two challenges arise: (i) estimate the target label marginal $p_t(y)$; and (ii) train a classifier f to maximize the performance on target domain. This paper focuses on the *relaxed label shift* setting. In particular, we assume that the label distribution can shift from source to target arbitrarily but that $p(x|y)$ varies between source and target in some comparatively subtle way. We keep this definition mathematically imprecise because we lack a rigorous characterization of the sense in which those shifts addressed in popular DA benchmarks are natural. While prior work addressing relaxed label shift has primarily focused on classifier performance, we also separately evaluate methods for estimating the target label distribution. This can be beneficial for two reasons. First, it can shed more light into how improving the estimates of target class proportion improves target performance. Second, understanding how the class proportions are changing can be of an independent interest.

A.1 Prior Work

Unsupervised domain adaption In our work, we focus on unsupervised DA where the goal is to adapt a predictor from a source distribution with labeled data to a target distribution from which we only observe unlabeled examples. Two popular settings for which DA is well-posed include (i) *covariate shift* [86, 84, 17, 16, 25] where $p(x)$ can change from source to target but $p(y|x)$ remains invariant; and (ii) *label shift* [58, 38, 3, 1, 23, 85] where the label marginal $p(y)$ can change but $p(x|y)$ is shared across source and target. Principled methods with strong theoretical guarantees exists for adaptation under these settings when target distribution's support is a subset of the source support. Ben-David et al. [6, 5], Mansour et al. [45], Zhao et al. [90], Wu et al. [80] present theoretical analysis when the assumptions of contained support is violated. More recently, a massive literature has emerged exploring a benchmark-driven heuristic approach [40, 41, 65, 67, 89, 88, 21, 64]. However, rigorous evaluation of popular DA methods is typically restricted to these carefully curated benchmark datasets where their is minor to no shift in label marginal from source to target.

Relaxed Label Shift Exploring the problem of shift in label marginal from source to target with natural variations in $p(x|y)$, a few papers highlighted theoretical and empirical failures of DA methods based on domain-adversarial neural network training [83, 80, 90]. Subsequently, several papers attempted to handle these problems in domain-adversarial training [68, 51, 39, 70, 44]. However, these methods often lack comparisons with other prominent DA methods and are evaluated under different datasets and model selection criteria. To this end, we perform a large scale rigorous comparison of prominent representative DA methods in a standardized evaluation framework.

Domain generalization In domain generalization, the model is given access to data from multiple different domains and the goal is to generalize to a previously unseen domain at test time [8, 47]. For a survey of different algorithms for domain generalization, we refer the reader to Gulrajani and Lopez-Paz [26]. A crucial distinction here is that unlike the domain generalization setting, in DA problems, we have access to unlabeled examples from the test domain.

Distinction from previous distribution shift benchmark studies Previous studies evaluating robustness under distribution shift predominantly focuses on transfer learning and domain generalization settings Wenzel et al. [78], Gulrajani and Lopez-Paz [26], Djolonga et al. [20], Wiles et al. [79], Koh et al. [34]. Taori et al. [71], Hendrycks et al. [30] studies the impact of robustness interventions (e.g. data augmentation techniques, adversarial training) on target (out of distribution) performance. Notably, Sagawa et al. [59] focused on evaluating DA methods on WILDS-2.0, an extended WILDS benchmark for DA setting. Our work is complementary to these studies, as we present the first extensive study of DA methods under shift in $p(y)$ and natural variations in $p(x|y)$.

B Future Work

In the future, we hope to extend RLSBENCH to cover natural language processing applications; tabular domains; and datasets from real applications in consequential domains such as healthcare and self-driving, where both shifts in label prevalences and perturbations in class conditional distributions can be expected across locations and over time. We also hope to incorporate self-supervised methods that learn representations by training on a union of unlabeled data from source and target via proxy tasks like reconstruction [24, 28] and contrastive learning [12, 14]. While re-weighting predictions using estimates of the target label distribution yields significant gains, the remaining gap between our results and oracle performance should motivate future work geared towards improved estimators. Also, we observe that the success of target label marginal estimation techniques depends on the nature of the shifts in $p(x|y)$. Mathematically characterizing the behavior of label shift estimation techniques when the label shift assumption is violated would be an important contribution.

C Main Results

Dataset	Source (w aug)	Source (adv)	BN-adapt	TENT	DANN	IW-DAN	CDAN	IW-CDAN	Fix-Match	Noisy-Student	Sentry
CIFAR-10	90.70	59.36	86.65	86.76	87.00	86.98	86.85	86.83	91.20	**92.15**	88.65
CIFAR-100	70.65	26.20	71.49	71.46	77.88	**78.51**	77.34	77.60	72.02	71.86	68.33
FMoW	60.11	49.51	56.77	58.02	57.79	57.09	57.36	57.16	60.36	**60.63**	49.62
Camelyon	75.21	81.27	86.64	87.33	81.17	82.21	84.41	85.17	**87.79**	85.99	87.39
Domainnet	52.88	48.93	53.42	54.08	51.83	52.04	54.00	54.14	**57.92**	54.36	50.48
Entity13	**81.50**	76.71	79.50	79.57	78.43	78.93	78.51	78.71	80.19	81.24	72.01
Entity30	69.82	60.92	68.45	68.49	65.78	66.07	64.75	64.62	**71.51**	69.75	57.00
Living17	74.50	49.27	71.56	71.17	68.52	71.98	70.24	69.91	**75.10**	74.62	54.32
Nonliving26	61.48	54.17	60.26	60.31	59.28	59.93	56.22	58.66	**62.20**	61.87	41.50
Officehome	64.59	59.08	65.67	65.57	66.51	66.59	66.48	66.32	64.77	**66.75**	58.51
Visda	59.76	55.74	67.18	68.43	68.21	67.94	71.04	70.63	73.50	61.10	**77.21**
Avg	69.20	56.47	69.78	70.11	69.31	69.84	69.75	69.98	**72.41**	70.94	64.09

Dirichlet Shift	Source (w aug)	Source (adv)	BN-adapt	TENT	DANN	IW-DAN	CDAN	IW-CDAN	Fix-Match	Noisy-Student	Sentry
∞ (NONE)	68.87	56.50	70.92	71.45	70.34	70.40	70.89	71.25	**73.58**	70.80	68.58
10.0	69.69	57.02	71.47	71.76	70.83	71.13	70.97	70.86	**73.73**	70.68	66.93
3.0	69.60	57.56	70.56	71.34	70.29	70.93	70.89	70.85	**73.89**	70.81	65.00
1.0	68.87	56.82	69.98	69.99	69.52	69.98	69.70	70.53	**72.76**	72.07	63.06
0.5	68.97	54.44	65.98	65.99	65.57	66.77	66.28	66.39	68.10	**70.33**	56.90
Avg	69.20	56.47	69.78	70.11	69.31	69.84	69.75	69.98	**72.41**	70.94	64.09

Table 1: *Results with different DA methods with source validation performance as early stopping criterion.* **(Top)** Aggregated across target label marginal shifts and **(Bottom)** aggregated across datasets and grouped by shift severity in label marginal. Smaller the Dirichlet shift parameter, more severe is the shift in target class proportion. While no single DA method performs consistently across different datasets, FixMatch seems to provide highest aggregate improvement over a source only classifier in our testbed. Moreover, shifts with $\alpha = \{10, 3.0, 1.0\}$ have little to no impact on different DA methods whereas performance of all DA methods degrade when $\alpha = 0.5$ falling below the performance of a source only classifier (except for Noisy Student). Parallel results with our meta algorithm included in Table 2. More detailed results with all methods on individual datasets in App. N.

Dataset	Source		BN-adapt				CDANN				FixMatch			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
CIFAR-10	90.7	**91.3**	86.7	89.8	90.7	**91.8**	86.9	88.1	87.1	**88.2**	91.2	92.4	92.1	**92.7**
CIFAR-100	**70.6**	69.2	71.5	71.6	**71.9**	71.6	77.3	**78.2**	77.2	77.8	72.0	71.3	**72.2**	71.7
FMoW	60.1	**60.9**	56.8	**57.5**	57.1	57.2	**57.4**	57.2	56.1	56.2	60.4	**60.8**	57.5	58.8
Camelyon	**75.2**	74.3	86.6	88.1	**88.8**	88.1	84.4	84.5	87.6	**88.1**	87.8	**88.5**	87.6	87.8
Domainnet	**52.9**	50.6	53.4	53.3	**53.6**	53.3	54.0	53.7	**54.8**	54.1	57.9	56.7	**58.4**	57.0
Entity13	81.5	**82.4**	79.5	80.7	81.0	**81.9**	78.5	**80.2**	77.3	78.8	80.2	81.6	82.3	**83.3**
Entity30	69.8	**70.9**	68.5	70.0	69.3	**70.9**	64.7	66.2	66.6	**68.6**	71.5	**72.7**	69.5	71.6
Living17	**74.5**	74.2	71.6	72.0	71.1	**72.9**	70.2	71.9	71.2	**72.5**	75.1	75.8	75.8	**76.9**
Nonliving26	61.5	**62.8**	60.3	62.1	61.9	**62.4**	56.2	58.0	58.7	**60.0**	62.2	61.9	62.9	**63.4**
Officehome	**64.6**	63.3	65.7	65.5	**65.9**	64.7	66.5	**66.6**	65.7	64.2	**64.8**	62.4	64.6	61.4
Visda	**59.8**	58.0	67.2	**68.7**	67.8	67.8	71.0	71.1	**74.3**	**74.3**	73.5	74.2	77.3	**77.7**
Avg	**69.2**	68.9	69.8	70.9	70.8	**71.2**	69.7	70.5	70.6	**71.2**	72.4	72.6	72.7	**72.9**

Dirichlet Shift	Source		BN-adapt				CDANN				FixMatch			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
∞ (NONE)	**68.9**	67.2	**70.9**	70.1	70.7	69.6	70.9	70.3	**71.0**	70.3	**73.6**	72.5	73.1	72.0
10.0	**69.7**	67.9	71.5	70.6	**71.5**	70.3	**71.0**	70.5	**71.0**	70.6	73.7	72.4	**74.3**	73.3
3.0	**69.6**	68.2	70.6	70.1	**70.9**	70.1	70.9	70.4	**71.4**	71.0	73.9	73.1	**74.0**	73.1
1.0	68.9	**69.8**	70.0	72.5	71.7	**72.8**	69.7	71.5	71.1	**72.5**	72.8	**74.0**	73.1	73.9
0.5	69.0	**71.5**	66.0	70.9	69.5	**72.9**	66.3	69.8	68.6	**71.5**	68.1	70.8	69.1	**72.3**
Avg	**69.2**	68.9	69.8	70.9	70.8	**71.2**	69.7	70.5	70.6	**71.2**	72.4	72.6	72.7	**72.9**

Table 2: *Results with BN-adapt, CDANN, and FixMatch with re-sampling (RS) and re-weighting (RW) correction (with RLLS estimate) with source validation performance as early stopping criterion.* **(Top)** Aggregated across target label marginal shifts and **(Bottom)** aggregated across datasets and grouped by shift severity in label marginal. Smaller the Dirichlet shift parameter, more severe is the shift in target class marginal. We boldface best correction result within each algorithm. RS and RW seem to help for all datasets and they both together significantly improve aggregate performance over no correction for all DA methods. While re-sampling consistently helps across different shifts, re-weighting hurts slightly when shift severity is small. However, for severe shifts in target label marginal ($\alpha \in \{1.0, 0.5\}$) re-weighting significantly improves performance. Parallel results with other methods in App. K.

Shift	Source	BN-adapt		TENT		DANN		CDANN		FixMatch		NoisyStudent	
	None	None	IS	None	RS	None	RS	None	RS	None	RS	None	RS
NONE	0.27	0.20	0.22	0.22	0.24	0.22	0.22	0.21	0.21	**0.20**	**0.20**	0.27	0.28
10.0	0.30	0.23	0.26	0.24	0.24	0.24	0.25	0.25	0.24	0.23	**0.22**	0.30	0.30
3.0	0.33	0.29	0.29	0.28	0.29	0.28	0.28	0.28	0.28	0.27	**0.25**	0.33	0.33
1.0	0.42	0.38	0.37	0.37	0.37	0.38	0.38	0.39	0.36	**0.35**	**0.35**	0.37	0.38
0.5	0.44	0.47	0.42	0.47	0.42	0.48	0.48	0.46	0.42	0.45	0.43	**0.40**	**0.40**
Avg	0.35	0.31	0.31	0.32	0.31	0.32	0.32	0.32	0.30	0.30	**0.29**	0.34	0.34

Table 3: *Target marginal estimation ℓ_1 error with RLLS across different DA methods aggregated grouped by shift severity in target label marginal.* Across all shift severities, RLLS with classifiers obtained with DA methods improves over RLLS with a source only classifier. Results with other estimation methods and across individual datasets in App. M.

D RLSbench Meta Algorithm

Algorithm 1 Meta algorithm to handle shift in class proportions

input Source training and validation data: (X_S, Y_S) and (X'_S, Y'_S), unlabeled target training and validation data: X_T and X'_T, classifier f, and DA algorithm $\mathcal{A}$

$\widetilde{X}_S, \widetilde{Y}_S \leftarrow \text{SampleClassBalanced}(X_S, Y_S)$ ▷ `Balance source data`
for $t = 1$ to T **do**
  $\widehat{Y}_T \leftarrow \arg\max_y f_y(X_T)$
  $\widetilde{X}_T \leftarrow \text{SampleClassBalanced}(X_T, \widehat{Y}_T)$ ▷ `Pseudo-balance target data`
  Run an epoch of $\mathcal{A}$ to update f on balanced source data $\{\widetilde{X}_S, \widetilde{Y}_S\}$ and target samples $\{\widetilde{X}_T\}$
end for
Estimate target marginal $\widehat{p}_t(y) \leftarrow \text{EstimateLabelMarginal}(f, X'_S, Y'_S, X'_T)$
$f'_j \leftarrow \dfrac{\widehat{p}_t(y=j) \cdot f_j}{\sum_k \widehat{p}_t(y=k) \cdot f_k}$ for all $j \in \mathcal{Y}$
▷ `Re-weight predictor with estimated label marginal`

output Target label marginal $\widehat{p}_t(y)$ and classifier f'

E Design choices in RLSbench

For a fair evaluation and comparison across different datasets and domain adaptation algorithms, we re-implemented all the algorithms with consistent design choices whenever applicable. We also make several additional implementation choices, described below. We defer the additional details to App. I.

Model selection criteria and hyperparameter choices Given that we lack validation i.i.d data from the target distribution, model selection in DA problems can not follow the standard workflow used in supervised training. Prior works often omit details on how to choose hyperparameters leaving open a possibility of choosing hyperparameters using the test set which can provide a false and unreliable sense of improvement. Moreover, inconsistent hyperparameter selection strategies can complicate fair evaluations misassociating the improvements to the algorithm under study.

In our work, we use source hold-out performance to pick the best hyperparameters. First, for ℓ_2 regularization and learning rate, we perform a sweep over random hyperparameters to maximize the performance of source only model on the hold-out source data. Then for each dataset, we keep these hyperparameters fixed across DA algorithms. For DA methods specific hyperparameters, we use the same hyperparameters across all the methods incorporating the suggestions made in corresponding papers. Within a run, we use hold out performance on source to pick the early stopping point. In appendices, we report *oracle* performance with choosing the early stopping point with target accuracy.

Evaluation criteria To evaluate the target label marginal estimation, we report ℓ_1 error between the estimated label distribution and true target label distribution. To evaluate the classifier performance on target data, we report performance of the (adapted) classifier on a hold-out partition of target data.

Architectural and pretraining details We experiment with different architectures (e.g., DenseNet121, Resenet18, Resnet50, ViT/B-16) across different datasets. We also experiment with CLIP-pretrained, Imagenet-pretrained, and randomly-initialized models. Given a dataset, for all experiments, we use the same architecture across different DA algorithms.

Data augmentation Data augmentation is a standard ingredient to train image classification models which can help approximate some of the variations between domains. Unless stated otherwise, we train all the methods using the standard strong augmentation technique: random horizontal flips, random crops of pre-defined size, augmentation with Cutout [19], and RandAugment [18]. To understand help with data augmentations alone in our setting, we also experiment with source only models trained without any data-augmentation.

F Dataset Details

In this section, we provide additional details about the datasets used in our benchmark study.

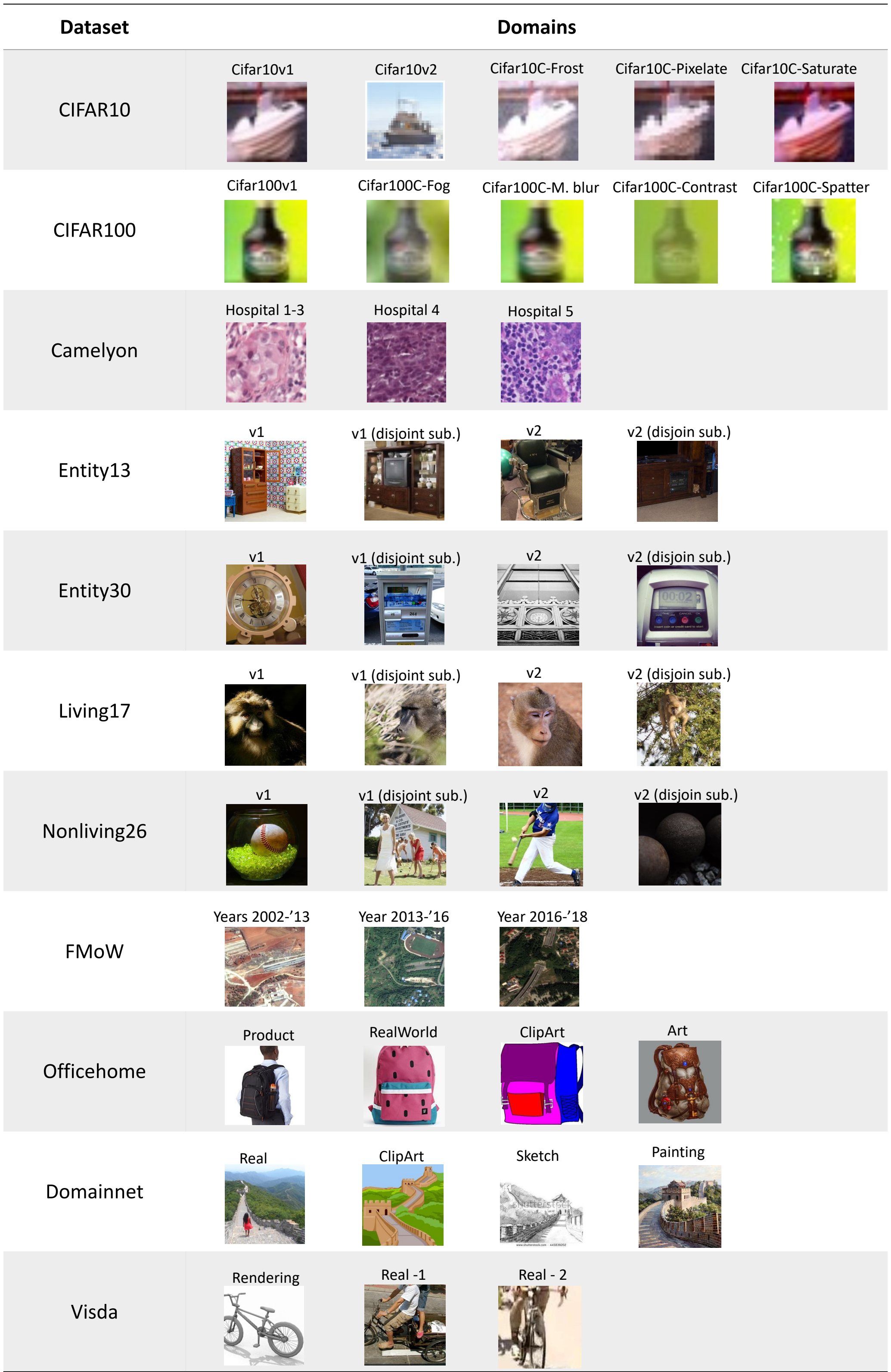

Figure 1: Examples from all the domains in each dataset.

- **CIFAR10** We use the original CIFAR10 dataset [35] as the source dataset. For target domains, we consider (i) synthetic shifts (CIFAR10-C) due to common corruptions [29]; and (ii) natural distribution shift, i.e., CIFAR10v2 [54, 73] due to differences in data collection strategy. We randomly sample 3 set of CIFAR-10-C datasets. Overall, we obtain 5 datasets (i.e., CIFAR10v1, CIFAR10v2, CIFAR10C-Frost (severity 4), CIFAR10C-Pixelate (severity 5), CIFAR10-C Saturate (severity 5)).
- **CIFAR100** Similar to CIFAR10, we use the original CIFAR100 set as the source dataset. For target domains we consider synthetic shifts (CIFAR100-C) due to common corruptions. We sample 4 CIFAR100-C datasets, overall obtaining 5 domains (i.e., CIFAR100, CIFAR100C-Fog (severity 4), CIFAR100C-Motion Blur (severity 2), CIFAR100C-Contrast (severity 4), CIFAR100C-spatter (severity 2)).
- **FMoW** In order to consider distribution shifts faced in the wild, we consider FMoW-WILDs [34, 15] from WILDS benchmark, which contains satellite images taken in different geographical regions and at different times. We use the original train as source and OOD val and OOD test splits as target domains as they are collected over different time-period. Overall, we obtain 3 different domains.
- **Camelyon17** Similar to FMoW, we consider tumor identification dataset from the wilds benchmark [4]. We use the default train as source and OOD val and OOD test splits as target domains as they are collected across different hospitals. Overall, we obtain 3 different domains.
- **BREEDs** We also considerBREEDs benchmark [60] in our setup to assess robustness to subpopulation shifts. BREEDs leverage class hierarchy in ImageNet to re-purpose original classes to be the subpopulations and defines a classification task on superclasses. We consider distribution shift due to subpopulation shift which is induced by directly making the subpopulations present in the training and test distributions disjoint. BREEDs benchmark contains 4 datasets **Entity-13**, **Entity-30**, **Living-17**, and **Non-living-26**, each focusing on different subtrees and levels in the hierarchy. We also consider natural shifts due to differences in the data collection process of ImageNet [57], e.g, ImageNetv2 [55] and a combination of both. Overall, for each of the 4 BREEDs datasets (i.e., Entity-13, Entity-30, Living-17, and Non-living-26), we obtain four different domains. We refer to them as follows: BREEDsv1 sub-population 1 (sampled from ImageNetv1), BREEDsv1 sub-population 2 (sampled from ImageNetv1), BREEDsv2 sub-population 1 (sampled from ImageNetv2), BREEDsv2 sub-population 2 (sampled from ImageNetv2). For each BREEDs dataset, we use BREEDsv1 sub-population A as source and the other three as target domains.
- **OfficeHome** We use four domains (art, clipart, product and real) from OfficeHome dataset [75]. We use the product domain as source and the other domains as target.
- **DomainNet** We use four domains (clipart, painting, real, sketch) from the Domainnet dataset [50]. We use real domain as the source and the other domains as target.
- **Visda** We use three domains (train, val and test) from the Visda dataset [49]. While 'train' domain contains synthetic renditions of the objects, 'val' and 'test' domains contain real world images. To avoid confusing, the domain names with their roles as splits, we rename them as 'synthetic', 'Real-1' and 'Real-2'. We use the synthetic (original train set) as the source domain and use the other domains as target.

Throughout the paper, we represent each multi-domain dataset with the name highlighted in the boldface above. Across these datasets, we obtain a total of 44 different source and target pairs. We also show example images in Fig. 1.

We provide scripts to setup these datasets with single command in our code. To investigate the performance of different methods under the stricter label shift setting, we also include a hold-out partition of source domain in the set of target domains. For these distribution shift pairs where source and target domains are i.i.d. partitions, we obtain the stricter label shift problem. We summarize the information about source and target domains in a table:

Train-test splits We partition each source and target dataset into 80% and 20% i.i.d. splits. We use 80% splits for training and 20% splits for evaluation (or validation). We throw away labels for the 80% target split and only use labels in the 20% target split for final evaluation. The rationale behind splitting the target data is to use a completely unseen batch of data for evaluation. This avoids evaluating on examples where a model potentially could have overfit. over-fitting to unlabeled examples for evaluation. In practice, if the aim is to make predictions on all the target data (i.e., transduction), we can simply use the (full) target set for training and evaluation.

Dataset	Source	Target
CIFAR10	CIFAR10v1	CIFAR10v1, CIFAR10v2, CIFAR10C-Frost (severity 4), CIFAR10C-Pixelate (severity 5), CIFAR10-C Saturate (severity 5)
CIFAR100	CIFAR100	CIFAR100, CIFAR100C-Fog (severity 4), CIFAR100C-Motion Blur (severity 2), CIFAR100C-Contrast (severity 4), CIFAR100C-spatter (severity 2)
Camelyon	Camelyon (Hospital 1–3)	Camelyon (Hospital 1–3), Camelyon (Hospital 4), Camelyon (Hospital 5)
FMoW	FMoW (2002–'13)	FMoW (2002–'13), FMoW (2013–'16), FMoW (2016–'18)
Entity13	Entity13 (ImageNetv1 sub-population 1)	Entity13 (ImageNetv1 sub-population 1), Entity13 (ImageNetv1 sub-population 2), Entity13 (ImageNetv2 sub-population 1), Entity13 (ImageNetv2 sub-population 2)
Entity30	Entity30 (ImageNetv1 sub-population 1)	Entity30 (ImageNetv1 sub-population 1), Entity30 (ImageNetv1 sub-population 2), Entity30 (ImageNetv2 sub-population 1), Entity30 (ImageNetv2 sub-population 2)
Living17	Living17 (ImageNetv1 sub-population 1)	Living17 (ImageNetv1 sub-population 1), Living17 (ImageNetv1 sub-population 2), Living17 (ImageNetv2 sub-population 1), Living17 (ImageNetv2 sub-population 2)
Nonliving26	Nonliving26 (ImageNetv1 sub-population 1)	Nonliving26 (ImageNetv1 sub-population 1), Nonliving26 (ImageNetv1 sub-population 2), Nonliving26 (ImageNetv2 sub-population 1), Nonliving26 (ImageNetv2 sub-population 2)
Officehome	Product	Product, Art, ClipArt, Real
DomainNet	Real	Real, Painiting, Sketch, ClipArt
Visda	Synthetic (originally referred to as train)	Synthetic, Real-1 (originally referred to as val), Real-2 (originally referred to as test)

Table 4: Details of the datasets considered in our RLSBENCH.

G Methods to estimate target marginal under the stricter label shift assumption

In this section, we describe the methods proposed to estimate the target label marginal under the stricter label shift assumption. Recall that under the label shift assumption, $p_s(y)$ can differ from $p_t(y)$ but the class conditional stays the same, i.e., $p_t(x|y) = p_s(x|y)$. We focus our discussion on recent methods that leverage off-the-shelf classifier. These approaches provide $\mathcal{O}(1/\sqrt{n})$ convergence rates under the label shift condition with mild assumptions on the classifier [38, 3, 23]. For simplicity, we assume we possess labeled source data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ and unlabeled target data $\{x_{n+1}, x_{n+2}, \ldots, x_{n+m}\}$.

While the relaxed label shift scenario violates the conditions required for consistency of label shift estimation techniques, we nonetheless employ these techniques and empirically evaluate efficacy of these methods in our testbed. In particular, to estimate the target label marginal, we experiment with: (i) RLLS [3]; (ii) MLLS [1]; and (iii) *baseline estimator* that simply averages the prediction of a classifier f on unlabeled target data.

RLLS First, we discuss *Regularized Learning under Label Shift* (RLLS) [3] (a variant of *Black Box Shift Estimation* (BBSE, Lipton et al. [38])): moment-matching based estimators that leverage (possibly biased, uncalibrated, or inaccurate) predictions to estimate the shift. RLLS solves the following optimization problem to estimate the importance weights $w_t(y) = \frac{p_t(y)}{p_s(y)}$ as:

$$\widehat{w}_t^{\text{RLLS}} = \underset{w \in \mathcal{W}}{\arg\min} \left\| \widehat{C}_f w - \widehat{\mu}_f \right\|_2 + \lambda_{\text{RLLS}} \left\| w - 1 \right\|_2 . \tag{1}$$

where $\mathcal{W} = \{w \in \mathbb{R}^d | \sum_y w(y) p_s(y) = 1 \text{ and } \forall y \in \mathcal{Y} \quad w(y) > 0\}$. $\widehat{C}_f$ is empirical confusion matrix of the classifier f on source data and $\widetilde{\mu}_f$ is the empirical average of predictions of the classifier f on unlabeled target data. With labeled source data data, the empirical confusion matrix can be computed as:

$$[\widehat{C}_f]_{i,j} = \frac{1}{n} \sum_{k=1}^{n} f_i(x_k) \cdot \mathbb{I}\left[y_k = j \right] .$$

To estimate target label marginal, we can multiple the estimated importance weights with the source label marginal (we can estimate source label marginal simply from labeled source data).

In our relaxed label shift problem, we use validation source data to compute the confusion matrix and use hold portion of target unlabeled data to compute μ_f. Unless specified otherwise, we use RLLS to estimate the target label marginal throughout the paper. We choose λ_{RLLS} as suggested in the original paper [3].

MLLS Next, we discuss Maximum Likelihood Label Shift (MLLS) [58, 1]: an Expectation Maximization (EM) algorithm that maximize the likelihood of observed unlabeled target data to estimate target label marginal assuming access to a classifier that outputs the source calibrated probabilities. In particular, MLLS uses the following objective:

$$\widehat{w}_t^{\text{MLLS}} = \underset{w \in \mathcal{W}}{\arg\min} \frac{1}{m} \sum_{i=1} \log(w^T f(x_{i+n})) , \tag{2}$$

where f is the classifier trained on source and $\mathcal{W}$ is the same constrained set defined above. We can again estimate the target label marginal by simply multiplying the estimated importance weights with the source label marginal.

Baseline estimator Given a classifier f, we can estimate the target label marginal as simply the average of the classifier output on unlabeled target data, i.e.,

$$\widehat{p}_t^{\text{baseline}} = \frac{1}{m} \sum_{i=1} f(x_{i+n}) . \tag{3}$$

Note that all of the methods discussed before leverage an off-the-shelf classifier f. Hence, we experiment with classifiers obtained with various deep domain adaptation heuristics to estimate the target label marginal.

Having obtained an estimate of target label marginal, we can simply re-weight the classifier with $\widehat{p}_t$ as $f'_j = \frac{\widehat{p}_t(y=j) \cdot f_j}{\sum_k \widehat{p}_t(y=k) \cdot f_k}$ for all $j \in \mathcal{Y}$. Note that, if we train f on a non-uniform source class-balance (and without re-balancing as in Step 1 of Algorithm 1), then we can re-weight the classifier with importance-weights $\widehat{w}_t$ as $f'_j = \frac{\widehat{w}_t(y=j) \cdot f_j}{\sum_k \widehat{w}_t(y=k) \cdot f_k}$ for all $j \in \mathcal{Y}$.

H Deep Domain Adapation methods

With the current version of RLSBENCH, we implement the following algorithms:

Source only As a baseline, we include model trained with empirical risk minimization [74] with cross-entropy loss on the source domain. We include source only models trained with and without augmentations. We also include adversarial robust models trained on source data with augmentations (**Source (adv)**). In particular, we use models adversarially trained against ℓ_2-perturbations.

Domain alignment methods These methods employ domain-adversarial training schemes aimed to learn invariant representations across different domains [21, 89, 70]. For our experiments, we include the following five representative methods: Domain Adversarial Neural Networks (**DANN** [21]), Conditional Domain Adversarial Neural Networks (**CDAN** [42], Importance-reweighted DANN (i.e., **IWDAN**) and CDAN (i.e., **IWCDAN**) proposed in Tachet des Combes et al. [69]).

Self-training methods These methods "pseudo-label" unlabeled examples with the model's own predictions and then train on them as if they were labeled examples. These methods often also use consistency regularization, which encourages the model to make consistent predictions on augmented views of unlabeled examples [36, 81, 7]. We include the following three algorithms: **FixMatch** [64], **Noisy Student** [81], Selective Entropy Optimization via Committee Consistency (**SENTRY** [51]).

Test-time adaptation methods take a source trained model and adapt few parameters (e.g. batch norm parameters, batch norm statistics) on the unlabeled target data with an aim to improve target performance. We include the following methods in our experimental suite: CORAL [66] or Domain Adjusted Regression (**DARE** [56]), BatchNorm adaptation (**BN-adapt** [37, 61]), Test entropy minimization (**TENT** [76]).

We now discuss each method in more detail and how it combines with our meta-algorithm to handle shift in class proportion.

H.1 Source only training

As a baseline, we consider empirical risk minimization on the labeled source data. Since this simply ignores the unlabeled target data, we call this as source only training. As mentioned in the main paper, we perform source only training with and without data augmentations. Formally, we minimize the following ERM loss:

$$L_{\text{source only}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(T(x_i), y_i)) \,, \tag{4}$$

where T is the stochastic data augmentation operation and ℓ is a loss function. Throughout the paper, we use cross-entropy loss minimization. Unless specified otherwise, we use strong augmentations as the data augmentation technique.

As mentioned in the main paper, we do not include re-sampling results with a source only model as it is trained only on source data and we observed no differences with just balancing the source data (as for most datasets source is already balanced) in our experiments. After obtaining a classifier f, we can first estimate the target label marginal and then adjust the classifier f with post-hoc re-weighting with importance ratios $w_t(y) = \widehat{p}_t(y)/\widehat{p}_s(y)$.

Adversarial training of a source only model Along with standard training of a source only model with data augmentation, we experiment with adversarially robust models [43]. To train adversarially robust models, we replace the standard ERM objective with a robust risk minimization objective:

$$L_{\text{source only (adv)}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(R(T(x_i), y_i), y_i) \,, \tag{5}$$

where $R(\cdot)$ performs the adversarial augmentation. In our paper, we use targeted Projected Gradient Descent (PGD) attacks with ℓ_2 perturbation model.

H.2 Domain-adversarial training methods

Domain-adversarial trianing methods seek to learn feature representations that are invariant across domains. These methods aimed at practical problems with non-overlapping support and are motivated by theoretical results showing that the gap between in- and out-of-distribution performance depends on some measure of divergence between the source and target distributions [5, 21]. While simultaneously minimizing the source error, these methods align the representations between source and target distribution. To perform alignment, these methods penalize divergence between feature representations across domains, encouraging the model to produce feature representations that are similar across domain.

Before describing these methods, we first define some notation. Consider a model $f = g \circ h$, where $h : \mathcal{X} \to \mathbb{R}^d$ is the featurizer that maps the inputs to some d dimensional feature space, and the head $g : \mathbb{R}^d \to \Delta^{k-1}$ maps the features to the prediction space. Following Sagawa et al. [59], with all of our domain invariant methods, we use strong augmentations with source and target data.

DANN DANN was proposed in Ganin et al. [21]. DANN approximates the divergence between feature representations of source and target domain by leveraging a domain discriminator classifier. Domain discriminator f_d aims to discriminate between source and target domains. Given a batch of inputs from source and target, this deep network f_d classifies whether the examples are from the source data or target data. In particular, the following loss function is used:

$$L_{\text{domain disc.}}(f_d) = \frac{1}{n}\sum_{i=1}^{n} \ell(f_d(h(T(x_i))), 0) + \frac{1}{n}\sum_{i=n+1}^{n+m} \ell(f_d(h(T(x_i))), 1)\,, \tag{6}$$

where $\{x_1, x_2, \ldots, x_n\}$ are n source examples and $\{x_{n+1}, \ldots, x_{m+n}\}$ are m target examples. Overall, the following loss function is used to optimize models with DANN:

$$L_{\text{DANN}}(h, g, f_d) = L_{\text{source only}}(g \circ h) - \lambda L_{\text{domain disc.}}(f_d)\,. \tag{7}$$

$L_{\text{DANN}}(h, g, f_d)$ is maximized with respect to the domain discriminator classifier and $L_{\text{DANN}}(h, g, f_d)$ minimized with respect to the underlying featurize and the source classifier. This is achieved by gradient reversal layer in practice. To train, three networks, we use three different learning rate η_f, η_g, and η_{f_d}. We discuss these hyperparameter details in App. I. We adapted our DANN implementation from Sagawa et al. [59] and Transfer learning library [33].

CDANN Conditional Domain adversarial neural network is a variant of DANN [42]. Here the domain discriminator is conditioned on the classifier g's prediction. In particular, instead of training the domain discriminator on the representation output of h, these methods operate on the outer product between the feature presentation $h(x)$ at an input x and the classifier's probabilistic prediction $f = g \circ h(x)$ (i.e., $h(x) \otimes f(x)$). Thus instead of training the domain discriminator classifier f_d on the d dimensional input space, they train it on $d \times k$ dimensional space. In particular, the following loss function is used:

$$L_{\text{CDAN domain disc.}}(f_d, g, h) = \frac{1}{n}\sum_{i=1}^{n} \ell(f_d(f \otimes h(T(x_i))), 0) + \frac{1}{n}\sum_{i=n+1}^{n+m} \ell(f_d(f \otimes h(T(x_i))), 1)\,, \tag{8}$$

where $\{x_1, x_2, \ldots, x_n\}$ are n source examples and $\{x_{n+1}, \ldots, x_{m+n}\}$ are m target examples. The overall loss is the same as DANN where $L_{\text{domain disc.}}(f_d)$ is replaced with $L_{\text{CDAN domain disc.}}(f_d, g, h)$.

We adapted our implementation for CDANN from Transfer learning library [33].

To adapt DANN and CDANN to our meta algorithm, at each epoch we can perform re-balancing of source and target data as in Step 1 and 4 of Algorithm 1. After obtaining the classifier f, we can use this classifier to first obtain an estimate of the target label marginal and then perform re-weighting adjustment with the obtained estimate.

IW-DANN and IW-CDANN Tachet et al. [68] proposed training with importance re-weighting correction with DANN and CDANN objectives to accommodate for the shift in the target label proportion. In particular, at every epoch of training they first estimate the importance ratio $\widehat{w}_t$ (with BBSE on training source and training target data) and then re-weight the domain discriminator objective and ERM objective. In particular, the domain discriminator loss for IW-DANN can be written as:

$$L^{\widehat{w}}_{\text{domain disc.}}(f_d) = \frac{1}{n}\sum_{i=1}^{n} \widehat{w}(y_i)\ell(f_d(h(T(x_i))), 0) + \frac{1}{n}\sum_{i=n+1}^{n+m} \ell(f_d(h(T(x_i))), 1)\,, \tag{9}$$

where we multiply the source loss with importance weights. Similarly, we can re-write the source only training objective with importance re-weighting as follows:

$$L^{\widehat{w}}_{\text{source only}}(f) = \frac{1}{n}\sum_{i=1}^{n} \widehat{w}(y_i)\ell(f(T(x_i), y_i))\,. \tag{10}$$

Overall, the following objective is used to optimize models with IW-DANN:

$$L_{\text{IW-DANN}}(h, g, f_d) = L^{\widehat{w}}_{\text{source only}}(g \circ h) - \lambda L^{\widehat{w}}_{\text{domain disc.}}(f_d)\,, \tag{11}$$

where the importance weights are updated after every epoch with classifier obtained in previous step. Similarly, with using importance re-weights with the CDANN objective, we obtain IW-CDANN objective.

In population, IW-CDANN and IW-DANN correction matches the correction with our meta-algorithm for DANN and CDANN. However, the behavior this importance re-weighting correction can be different from our meta-algorithm for over-parameterized models with finite data [10]. Recent empirical and theoretical findings have highlighted that importance re-weighting have minor to no effect on overparameterized models when trained for several epochs [10, 82]. On the other hand, with finite samples, re-sampling (when class labels are available) has shown different and promising empirical behavior [2, 32]. This may highlight the differences in the behavior of IW-CDANN (or IW-DANN) with our meta algorithm on CDANN (or DANN).

We refer to the implementation provided by the authors [68].

H.3 Self-training methods

Self-training methods leverage unlabeled data by 'pseudo-labeling' unlabeled examples with the classifier's own predictions and training on them as if they were labeled examples. Recent self-training methods also often make use of consistency regularization, for example, encouraging the model to make similar predictions on augmented versions of unlabeled example. In our work, we experiment with the following methods:

FixMatch Sohn et al. [64] proposed FixMatch as a variant of the simpler Pseudo-label method [36]. This algorithm dynamically generates psuedolabels and overfits on them in each batch. FixMatch employs consistency regularization on the unlabeled data. In particular, while pseudolabels are generated on a weakly augmented view of the unlabeled examples, the loss is computed with respect to predictions on a strongly augmented view. The intuition behind such an update is encourage a model to make predictions on weakly augmented data consistent with the strongly augmented example. Moreover, FixMatch only overfits to the assigned labeled with weak-augmentation if the confidence of the prediction with strong augmentation is greater than some threshold τ.

Refer to T_{weak} as the weak-augmentation and T_{strong} as the strong-augmentation function. Then, FixMatch uses the following loss function:

$$\begin{aligned} L_{\text{FixMatch}}(f) = &\frac{1}{n}\sum_{i=1}^{n} \ell(f(T_{\text{strong}}(x_i), y_i)) \\ &+ \frac{\lambda}{m}\sum_{i=n+1}^{m+n} \ell(f(T_{\text{strong}}(x_i), \widetilde{y}_i)) \cdot \mathbb{I}\left[\max_y f_y(T_{\text{strong}}(x_i)) \geqslant \tau\right], \end{aligned}$$

where $\widetilde{y}_i = \arg\max_y f_y(T_{\text{weak}}(x_i))$. We adapted our implementation from Sagawa et al. [59] which matches the implementation of Sohn et al. [64] except for one detail. While Sohn et al. [64] augments labeled examples with weak augmentation, Sagawa et al. [59] proposed to strongly augment the labeled source examples.

NoisyStudent Xie et al. [81] proposed a different variant of Pseudo-labeling. Unlike FixMatch, Noisy Student generates pseudolabels, fixes them, and then trains the model until convergence before generating new pseudolabels. The first set of pseudolabels are obtained with training an initial teacher model only on the source labeled data. Then in each iteration, a randomly initialized models fits to the labeled source data and pseudolabeled target data with pseudolabels assigned the converged model in the previous iteration. Noisy student objective can be summarized as:

$$L_{\text{NoisyStudent}}(f^N) = \frac{1}{n}\sum_{i=1}^{n} \ell(f^N(T_{\text{strong}}(x_i), y_i)) + \frac{1}{m}\sum_{i=n+1}^{m+n} \ell(f^N(T_{\text{strong}}(x_i), \widetilde{y}_i)),$$

where $\widetilde{y}_i = \arg\max_y f_y^{N-1}(T_{\text{weak}}(x_i))$ is computed with the classifier obtained at $N-1$ step. Note that the randomly initialized model at each iteration uses a dropout of $p = 0.5$ in the penultimate layer. We adapted our implementation of NoisyStudent to Sagawa et al. [59]. To initialize the initial teacher model, we use the source-only model trained with strong augmentations without dropout.

SENTRY Prabhu et al. [51] proposed a different variant of pseudolabeling method. This method is aimed to tackle DA under relaxed label shift scenario. a SENTRY incorporates a target instance based on its predictive consistency under a committee of strong image transformations. In particular, SENTRY makes N strong augmentations of an unlabeled target example and makes a prediction on those. If the majority of the committee matches the prediction on the sample example with weak-augmentation then entropy is minimized on that example, otherwise the entropy is maximized. Moreover, the authors employ an 'information-entropy' objective aimed to match the prediction at every example with the estimated target label marginal. Overall the SENTRY objective is defined as follows:

$$L_{\text{SENTRY}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(T_{\text{strong}}(x_i), y_i)) + \frac{1}{m}\sum_{i=n+1}^{m+n}\sum_{j=1}^{k} f_k(y=j|x_i)\log(\widetilde{p}_t(y=j))$$
$$+ \lambda_{\text{unsup}}\frac{1}{m}\sum_{i=n+1}^{m+n}\sum_{j=1}^{k} -f_k(y=j|x_i)\log(f_k(y=j|x_i))\cdot(2l(x)-1)\,,$$

where $l(x) \in \{0, 1\}$ is majority vote output of the committee consistency. For more details, we refer the reader to Prabhu et al. [51]. Additionally, at each training epoch, SENTRY balances the source data and pseudo-balances the target data. We adapted our implementation with the official implementation in Prabhu et al. [51] with minor differences.

H.4 Test-time training methods

These take a already trained source model and adapt few parameters (e.g. batch norm parameters, batch norm statistics) on the unlabeled target data with an aim to improve target performance. These methods are computationally cheaper than other DA methods in suite as they adapt a classifier on-the-fly. We include the following methods in our experimental suite:

DARE Sun et al. [66] proposed CORAL to adapt a model trained on source to target by whitening the feature representations. In particular, say $\widehat{\Sigma}_s$ is the empirical covariance of the target data representations and Σ_s is the empirical covariance of the source data representations, CORAL adjusts a linear layer g on target by re-training the final layer on the outputs: $\Sigma_t^{1/2}\Sigma_s^{-1/2}h(x)$. DARE [56] simplified the procedure and showed that this is equivalent to training a linear head h on $\Sigma_s^{-1/2}h(x)$ and whitening target data representations with $\Sigma_t^{-1/2}h(x)$ before input to the classifier. We choose to implement the latter procedure as it is cheap to train a single classifier in multi-domain datasets.

BN-adapt Li et al. [37] proposed batch norm adaptation. More recently, Schneider et al. [61] showed gains with BN-adapt on common corruptions benchmark. Batch norm adaptation is applicable for deep models with batch norm parameters. With this method we simply adapt the Batchnorm statistics, in particular, mean and std of each batch norm layer.

TENT Wang et al. [76] proposed optimizing batch norm parameters to minimize entropy of the predictor on the unlabeled target data. In our implementation of TENT, we perform BN-adapt before learning batch norm parameters.

With our meta algorithm, before adapting the source only classifier with test time adaptation methods, we use it to perform the re-sampling correction. After obtaining the adapted classifier, we estimate target label marginal and use it to adjust the classifier with re-weighting.

I Hyperparameter and Architecture Details

I.1 Architecture and Pretraining Details

For all datasets, we used the same architecture across different algorithms:

- CIFAR-10: Resnet-18 [27] pretrained on Imagenet
- CIFAR-100: Resnet-18 [27] pretrained on Imagenet

- Camelyon: Densenet-121 [31] *not* pretrained on Imagenet as per the suggestion made in [34]
- FMoW: Densenet-121 [31] pretrained on Imagenet
- BREEDs (Entity13, Entity30, Living17, Nonliving26): Resnet-18 [27] *not* pretrained on Imagenet as per the suggestion in [60]. The main rationale is to avoid pre-training on the superset dataset where we are simulating sub-population shift.
- Officehome: Resnet-50 [27] pretrained on Imagenet
- Domainnet: Resnet-50 [27] pretrained on Imagenet
- Visda: Resnet-50 [27] pretrained on Imagenet

Except for Resnets on CIFAR datasets, we used the standard pytorch implementation [22]. For Resnet on cifar, we refer to the implementation here: `https://github.com/kuangliu/pytorch-cifar`. For all the architectures, whenever applicable, we add antialiasing [87]. We use the official library released with the paper.

For imagenet-pretrained models with standard architectures, we use the publicly available models here: `https://pytorch.org/vision/stable/models.html`. For imagenet-pretrained models on the reduced input size images (e.g. CIFAR-10), we train a model on Imagenet on reduced input size from scratch. We include the model with our publicly available repository.

In our work, we also experiment with CLIP pre-training [53]. In particular, we experiment with VIT-B16 model. We include clip results in App. N.

I.2 Hyperparameters

First, we tune learning rate and ℓ_2 regularization parameter by fixing batch size for each dataset that correspond to maximum we can fit to 15GB GPU memory. We set the number of epochs for training as per the suggestions of the authors of respective benchmarks. Note that we define the number of epochs as a full pass over the labeled training source data. We summarize learning rate, batch size, number of epochs, and ℓ_2 regularization parameter used in our study in Table 5.

Dataset	Epoch	Batch size	ℓ_2 regularization	Learning rate
CIFAR10	50	200	0.001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0001 (chosen from $\{0.0, 0.001, 0.01, 0.0001\}$)
CIFAR100	50	200	0.001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0001 (chosen from $\{0.0, 0.001, 0.01, 0.0001\}$)
Camelyon	10	96	0.003 (chosen from $\{0.003, 0.03, 0.0003\}$)	0.01 (chosen from $\{0.0, 0.1, 0.001, 0.01\}$)
FMoW	30	64	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0 (chosen from $\{0.0, 0.001, 0.01, 0.0001\}$)
Entity13	40	256	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01\}$)	5e-5 (chosen from {5e-5, 5e-4, 1e-4, 1e-5})
Entity30	40	256	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01\}$)	5e-5 (chosen from {5e-5, 5e-4, 1e-4, 1e-5})
Living17	40	256	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01\}$)	5e-5 (chosen from {5e-5, 5e-4, 1e-4, 1e-5})
Nonliving26	40	256	0.2 (chosen from $\{0.1, 0.5, 0.2, 0.01\}$)	5e-5 (chosen from {5e-5, 5e-4, 1e-4, 1e-5})
Officehome	50	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0001 (chosen from $\{0.0005, 0.001, 0.0001\}$)
DomainNet	15	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0001 (chosen from $\{0.0005, 0.001, 0.0001\}$)
Visda	10	96	0.0001 (chosen from $\{0.0001, 0.001,$1e-5$\}$)	0.0005 (chosen from $\{0.0005, 0.001, 0.0001\}$)

Table 5: Details of the learning rate and batch size considered in our RLSBENCH

For each algorithm, we use the hyperparameters reported in the initial papers. For domain-adversarial methods (DANN and CDANN), we refer to the suggestions made in Transfer Learning Library [33]. We tabulate hyperparameters for each algorithm next:

- **DANN, CDANN, IW-CDANN and IW-DANN** As per Transfer Learning Library suggestion, we use a learning rate multiplier of 0.1 for the featurizer. We default to a penalty weight of 1.0 for all datasets with pre-trained initialization. For BREEDs and camelyon, we default to a penalty weight of 0.1 as we do not use a pre-trained architecture.
- **FixMatch** We use the lambda is 1.0 and use threshold tau as 0.1.
- **NoisyStudent** We repeat the procedure for 2 iterations and use a drop level of $p = 0.5$.
- **SENTRY** We use $\lambda_{\text{src}} = 1.0$, $\lambda_{\text{ent}} = 1.0$, and $\lambda_{\text{unsup}} = 0.1$. We use a committee of size 3.

I.3 Compute Infrastructure

Our experiments were performed across a combination of Nvidia T4, A6000, P100 and V100 GPUs. Overall, to run the entire RLSBENCH suite on a T4 GPU machine with 8 CPU cores we would approximately need $70k$ GPU hours of compute.

I.4 Data Augmentation

In our experiments, we leverage data augmentation techniques that encourage robustness to some variations between domains.

For weak augmentation, we leverage random horizontal flips and random crops of pre-defined size. For strong augmentation, we apply the following transformations sequentially: random horizontal flips, random crops of pre-defined size, augmentation with Cutout [19], and RandAugment [18]. For the exact implementation of RandAugment, we directly use the implementation of Sohn et al. [64]. The pool of operations includes: autocontrast, brightness, color jitter, contrast, equalize, posterize, rotation, sharpness, horizontal and vertical shearing, solarize, and horizontal and vertical translations. We apply N = 2 random operations for all experiments.

J Comparison with SENTRY on officehome dataset with different hyperparameters

On the Officehome dataset, we observe a slight discrepancy between SENTRY results with our runs and numbers originally reported in the paper [51]. We observe significant improvements with FixMatch over SENTRY. However, in the original paper, SENTRY outperformed FixMatch on Officehome. We find that this discrepancy is due to differences in batch size used in original work versus in our runs (which we kept same for all the algorithms). In this section, we report SENTRY results with the updated batch size. With the new batch size, we reconcile SENTRY results but also observe a significant improvement in FixMatch results.

We note that for the main experiments on Officehome dataset, we used a batch size of 96 for all methods including SENTRY. However, SENTRY reported results with a batch size of 16 in their work. Hence, we re-run the SENTRY algorithm with a batch size of 16. To investigate the impact of the decreased batch size, we make a comparison with FixMatch (the best algorithm on Officehome in our runs) by re-running it with the decreased batch size.

In Table 6 we report results on individual shift pairs in officehome. We observe that SENTRY improves over FixMatch for the default minor shift in the label distribution in the officehome dataset. However, as the shift severity increases we observe that SENTRY performance degrades. Overall, we observe that RS-FixMatch performs similar or superior to SENTRY on 3 out of 4 shift pairs in officehome.

More generally, across our runs, we also observed model training with SENTRY to be unstable. Investigating further, we observe that the maximization objective to enforce consistency cause instabilities. This behavior is specifically prevalent for experiments where we don't use initiale the underlying model with pre-trained weights (for example, in BREEDs datasets).

Algorithm	Alpha = None	Alpha = 10.0	Alpha = 3.0	Alpha = 1.0	Alpha = 0.5	Avg
FixMatch	92.5	95.2	98.0	100.0	100.0	97.1
RS-FixMatch	92.5	96.4	98.0	100.0	100.0	97.4
SENTRY	93.0	94.0	98.0	83.3	87.5	91.2

(a) Product to Product (in-distribution)

Algorithm	Alpha = None	Alpha = 10.0	Alpha = 3.0	Alpha = 1.0	Alpha = 0.5	Avg
FixMatch	71.4	71.5	70.7	73.1	75.5	72.4
RS-FixMatch	74.7	74.0	72.1	73.1	70.4	72.9
SENTRY	78.1	78.0	75.1	71.7	65.3	73.6

(b) Product to Real

Algorithm	Alpha = None	Alpha = 10.0	Alpha = 3.0	Alpha = 1.0	Alpha = 0.5	Avg
FixMatch	41.5	44.0	44.2	48.4	39.4	43.5
RS-FixMatch	45.5	44.8	43.6	50.0	37.4	44.2
SENTRY	45.8	46.5	41.4	40.3	27.3	40.3

(c) Product to ClipArt

Algorithm	Alpha = None	Alpha = 10.0	Alpha = 3.0	Alpha = 1.0	Alpha = 0.5	Avg
FixMatch	54.4	51.3	54.7	57.3	55.9	54.7
RS-FixMatch	57.2	53.6	55.9	57.3	58.8	56.6
SENTRY	63.7	62.0	62.1	65.3	55.9	61.8

(d) Product to Art

Table 6: Officehome results with batch size 16 instead of 96 used throughout our experiments.

K Results with RW and RS for DANN, TENT and Noisy-Student

Dataset	**TENT**				**DANN**				**NoisyStudent**			
	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
cifar10	86.8	89.9	90.7	91.8	87.0	88.2	85.6	85.5	92.2	92.3	92.2	92.3
cifar100	71.5	71.6	71.9	71.6	77.9	79.4	76.6	77.5	71.9	71.0	71.9	71.0
fmow	58.0	58.2	57.8	57.8	57.8	57.9	56.8	56.6	60.6	61.1	61.0	60.6
camelyon	87.3	88.5	89.4	90.4	81.2	80.9	80.4	79.8	86.0	86.0	86.4	86.4
domainnet	54.1	54.2	54.4	54.2	51.8	51.8	53.5	53.2	54.4	52.4	54.3	51.9
entity13	79.6	80.8	81.0	81.9	78.4	79.5	78.6	79.8	81.2	82.1	81.6	82.8
entity30	68.5	70.1	69.3	70.9	65.8	66.9	65.4	66.9	69.7	70.0	69.4	70.7
living17	71.2	71.9	71.1	72.9	68.5	71.3	70.5	71.5	74.6	74.3	71.0	75.9
nonliving26	60.3	62.1	61.9	62.4	59.3	60.7	56.7	56.5	61.9	62.3	62.7	63.3
officehome	65.6	65.8	65.8	64.9	66.5	66.6	67.7	66.7	66.7	64.7	66.8	64.6
visda	68.4	69.9	68.7	68.8	68.2	68.3	71.9	72.1	61.1	59.7	61.2	59.5
Avg	70.1	71.2	71.1	**71.6**	69.3	**70.2**	69.4	69.7	**70.9**	70.5	**70.8**	**70.8**

Table 7: *Results with TENT, DANN, and NoisyStudent with re-sampling and re-weighting correction with source validation performance as early stopping criterion aggregated across target label marginal shifts*. Re-sampling and Re-weighting seem to help for all datasets and they both together improve aggregate performance over no correction for all DA methods.

Dataset	**TENT**				**DANN**				**NoisyStudent**			
	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
∞(NONE)	71.5	70.3	71.4	69.9	70.3	69.6	70.2	69.4	70.8	69.4	70.7	69.0
10.0	71.8	70.7	72.1	70.8	70.8	70.2	70.3	69.6	70.7	69.4	71.1	69.6
3.0	71.3	70.6	71.5	70.4	70.3	70.4	71.0	70.3	70.8	69.6	70.7	69.7
1.0	70.0	72.0	71.3	72.5	69.5	71.1	69.8	70.8	72.1	72.2	71.6	72.4
0.5	66.0	70.6	69.2	72.8	65.6	69.5	65.7	68.2	70.3	72.1	69.8	73.4
Avg	70.1	70.8	71.1	**71.3**	69.3	**70.2**	69.4	69.7	**70.9**	70.5	**70.8**	**70.8**

Table 8: *Results with TENT, DANN, NoisyStudent with re-sampling and re-weighting correction with source validation performance as early stopping criterion grouped by shift severity*. Re-sampling performs similar or helps across different shifts whereas re-weighting hurts slightly when shift severity is small. However, for severe shifts in target label marginal ($\alpha \in \{1.0, 0.5\}$) re-weighting significantly improves performance.

L Results with Oracle Early Stopping Criterion

In this section, we report results with oracle early stopping criterion. We observe differences in performance when using target performance versus source hold-out performance for model selection. This highlights a more nuanced behavior than the accuracy-on-the-line phenomena [46, 55]. We hope to study this contrasting behavior in more detail in future work.

L.1 Results with target validation performance for all methods WITHOUT re-sampling and re-weighting correction

Dataset	Source (w aug)	Source (adv)	BN-adapt	TENT	DANN	IW-DAN	CDAN	IW-CDAN	Fix-Match	Noisy-Student	Sentry
cifar10	91.02	59.36	87.11	87.12	87.47	87.50	87.45	87.49	91.62	92.43	89.18
cifar100	71.38	26.20	72.04	72.05	78.84	79.37	78.30	78.35	72.58	72.46	69.05
fmow	60.89	49.51	57.52	58.73	58.75	58.69	58.56	58.46	61.42	62.27	49.97
camelyon	87.26	81.27	89.93	89.30	83.61	83.72	88.95	88.33	90.02	87.84	89.32
domainnet	53.35	48.93	53.77	54.41	53.52	53.59	54.91	54.86	58.20	55.01	51.03
entity13	81.86	76.71	80.22	80.28	80.01	80.24	80.28	79.71	82.62	82.52	73.47
entity30	70.72	60.92	69.75	69.80	66.98	67.65	66.76	67.38	72.95	70.70	58.61
living17	78.56	49.27	76.94	76.75	77.23	75.12	75.54	75.33	78.80	77.41	61.05
nonliving26	65.24	54.17	63.93	63.95	61.87	62.90	60.51	61.08	66.69	65.50	45.86
officehome	66.23	59.08	66.79	66.78	69.00	69.29	69.31	69.33	66.47	68.75	60.48
visda	63.97	55.74	68.52	69.58	73.42	73.82	76.50	76.96	78.21	62.64	80.16
Avg	71.86	56.47	71.50	71.70	71.88	71.99	72.46	72.48	74.51	72.50	66.20

Table 9: Results with different DA methods with target validation performance as early stopping criterion aggregated across target label marginal shifts.

Shift	Source (w aug)	Source (adv)	BN-adapt	TENT	DANN	IW-DAN	CDAN	IW-CDAN	Fix-Match	Noisy-Student	Sentry
100.0	70.43	56.50	71.84	72.16	71.12	71.25	71.62	71.85	74.61	71.57	69.16
10.0	71.24	57.02	72.37	72.75	71.60	71.98	72.60	72.36	75.01	71.81	67.87
3.0	71.37	57.56	72.19	72.48	71.94	72.13	72.57	72.46	75.49	72.50	66.65
1.0	73.19	56.82	72.44	72.46	72.96	72.89	73.21	73.85	75.19	73.73	66.05
0.5	73.09	54.44	68.67	68.67	71.79	71.70	72.29	71.87	72.24	72.91	61.25
Avg	71.86	56.47	71.50	71.70	71.88	71.99	72.46	72.48	74.51	72.50	66.20

Table 10: Results with different DA methods with target validation performance as early stopping criterion aggregated across datasets and grouped by shift severity in target label marginal.

L.2 Results with target validation performance for all methods WITH re-sampling and re-weighting correction

Dataset	Source		BN-adapt				CDANN				FixMatch			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
cifar10	91.0	91.7	87.1	90.4	91.3	92.3	87.4	88.5	87.7	88.6	91.6	92.7	92.4	93.0
cifar100	71.4	70.0	72.0	72.2	72.6	72.4	78.3	79.1	77.8	78.7	72.6	71.9	72.6	72.3
fmow	60.9	61.5	57.5	58.6	58.1	58.6	58.6	58.6	56.9	57.2	61.4	62.4	58.3	60.2
camelyon	87.3	88.5	89.9	90.9	91.6	90.4	88.9	89.5	89.1	89.5	90.0	90.8	90.4	91.6
domainnet	53.4	50.9	53.8	53.8	54.3	54.0	54.9	54.9	55.3	55.0	58.2	57.0	58.6	57.4
entity13	81.9	82.6	80.2	81.4	81.7	82.9	80.3	81.5	78.8	79.9	82.6	84.0	83.6	84.6
entity30	70.7	72.2	69.8	71.5	70.7	72.0	66.8	68.9	68.0	70.2	73.0	74.2	72.2	73.8
living17	78.6	77.8	76.9	76.5	79.3	77.3	75.5	76.6	75.4	76.2	78.8	81.5	81.0	81.0
nonliving26	65.2	66.7	63.9	65.5	65.8	65.7	60.5	62.2	60.4	61.7	66.7	68.5	67.2	68.1
officehome	66.2	65.0	66.8	66.9	67.1	66.6	69.3	69.3	69.1	69.1	66.5	63.7	66.2	63.2
visda	64.0	61.8	68.5	70.3	69.9	70.2	76.5	76.9	78.0	78.4	78.2	79.0	80.7	81.1
Avg	71.9	71.7	71.5	72.6	72.9	72.9	72.5	73.3	72.4	73.1	74.5	75.1	74.9	75.1

Table 11: Results with BN-adapt, CDANN, and FixMatch with re-sampling and re-weighting correction with target validation performance as early stopping criterion aggregated across target label marginal shifts.

Shift	Source		BN-adapt				CDANN				FixMatch			
	None	RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
NONE	70.4	68.9	71.8	71.1	71.7	70.6	71.6	71.1	71.6	70.9	74.6	73.6	74.4	73.4
10.0	71.2	69.7	72.4	71.8	72.4	71.5	72.6	71.8	72.3	71.8	75.0	73.9	75.2	74.2
3.0	71.4	70.1	72.2	71.8	72.7	71.6	72.6	72.1	72.8	72.4	75.5	74.8	75.3	74.4
1.0	73.2	73.6	72.4	74.3	74.1	74.8	73.2	74.9	73.4	75.2	75.2	76.3	76.0	76.8
0.5	73.1	76.2	68.7	73.8	73.8	76.1	72.3	76.4	71.8	75.3	72.2	76.8	73.4	76.7
Avg	71.9	71.7	71.5	72.6	72.9	72.9	72.5	73.3	72.4	73.1	74.5	75.1	74.9	75.1

Table 12: Results with BN-adapt, CDANN, and FixMatch with re-sampling and re-weighting correction with target validation performance as early stopping criterion grouped by shift severity.

Dataset	TENT				DANN				NoisyStudent			
	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
cifar10	87.1	90.4	91.3	92.3	87.5	88.6	86.0	85.9	92.4	92.6	92.4	92.6
cifar100	72.0	72.2	72.6	72.4	78.8	80.0	77.7	78.5	72.5	71.6	72.4	71.5
fmow	58.7	58.5	59.0	57.8	58.8	59.1	57.3	57.6	62.3	62.6	62.1	62.0
camelyon	89.3	91.2	92.2	91.3	83.6	83.7	83.1	82.8	87.8	88.1	88.3	88.3
domainnet	54.4	53.8	55.0	54.2	53.5	53.6	54.9	54.5	55.0	52.8	54.8	52.5
entity13	80.3	81.5	81.8	82.9	80.0	81.0	79.7	80.7	82.5	83.3	82.6	83.6
entity30	69.8	71.5	70.7	72.0	67.0	68.5	67.5	69.8	70.7	72.1	71.2	72.5
living17	76.7	76.3	79.3	77.3	77.2	77.1	76.4	77.0	77.4	80.0	79.5	79.2
nonliving26	63.9	65.5	65.8	65.7	61.9	63.0	60.4	61.8	65.5	66.1	65.7	65.1
officehome	66.8	65.8	67.2	65.7	69.0	69.0	69.7	69.1	68.8	66.2	68.7	66.2
visda	69.6	70.9	70.7	70.5	73.4	74.3	75.5	76.1	62.6	61.2	62.5	60.7
Avg	71.7	72.5	73.2	72.9	71.9	72.5	71.7	72.2	72.5	72.4	72.7	72.2

Table 13: *Results with TENT, DANN, and NoisyStudent with re-sampling and re-weighting correction with target validation performance as early stopping criterion aggregated across target label marginal shifts*.

Dataset	TENT				DANN				NoisyStudent			
	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW	None	RW	RS	RS+ RW
NONE	72.2	71.0	72.2	70.8	71.1	70.4	70.8	70.0	71.6	70.1	71.7	69.9
10.0	72.8	71.8	72.9	71.7	71.6	71.3	71.7	70.8	71.8	70.5	72.2	70.6
3.0	72.5	71.8	73.0	71.7	71.9	71.7	72.2	71.5	72.5	70.9	72.4	71.0
1.0	72.5	74.2	74.1	74.6	73.0	73.9	72.4	73.7	73.7	74.2	73.7	73.9
0.5	68.7	73.8	74.0	75.8	71.8	75.4	71.2	74.7	72.9	76.4	73.7	75.6
Avg	71.7	72.5	73.2	72.9	71.9	72.5	71.7	72.2	72.5	72.4	72.7	72.2

Table 14: Results with TENT, DANN, NoisyStudent with re-sampling and re-weighting correction with target validation performance as early stopping criterion grouped by shift severity.

M Target Marginal Estimation

Shift	Source	BN-adapt		TENT		DANN		CDANN		FixMatch		NoisyStudent	
	None	None	IS	None	IS	None	IS	None	IS	None	IS	None	IS
cifar10	0.08	0.11	0.07	0.11	0.07	0.10	0.13	0.11	0.10	0.07	0.05	0.05	0.05
cifar100	0.33	0.29	0.28	0.29	0.29	0.22	0.23	0.22	0.23	0.29	0.28	0.32	0.32
fmow	0.33	0.37	0.39	0.45	0.46	0.39	0.40	0.42	0.43	0.32	0.37	0.33	0.33
camelyon	0.39	0.20	0.23	0.16	0.19	0.27	0.31	0.19	0.10	0.13	0.18	0.19	0.19
domainnet	0.68	0.56	0.57	0.55	0.56	0.60	0.56	0.59	0.57	0.52	0.51	0.68	0.68
entity13	0.12	0.15	0.13	0.15	0.13	0.14	0.13	0.14	0.14	0.15	0.13	0.13	0.13
entity30	0.28	0.28	0.27	0.28	0.27	0.31	0.31	0.31	0.29	0.27	0.28	0.27	0.28
living17	0.33	0.34	0.33	0.34	0.33	0.38	0.34	0.37	0.35	0.34	0.31	0.35	0.36
nonliving26	0.40	0.41	0.40	0.41	0.40	0.38	0.43	0.44	0.40	0.43	0.41	0.38	0.40
officehome	0.48	0.44	0.45	0.44	0.45	0.44	0.43	0.45	0.45	0.48	0.47	0.49	0.49
visda	0.58	0.37	0.40	0.36	0.39	0.38	0.33	0.36	0.29	0.35	0.26	0.60	0.59
Avg	0.36	0.32	0.32	0.32	0.32	0.33	0.33	0.33	0.31	0.30	0.30	0.35	0.35

Table 15: Target marginal estimation ℓ_1 error with RLLS across different DA methods aggregated across different target label marginal shifts for different datasets.

Shift	Source	BN-adapt		TENT		DANN		CDANN		FixMatch		NoisyStudent	
	None	None	IS	None	IS	None	IS	None	IS	None	IS	None	IS
None	0.21	0.15	0.16	0.16	0.17	0.17	0.17	0.17	0.16	0.16	0.15	0.22	0.22
10.0	0.25	0.19	0.20	0.20	0.20	0.20	0.20	0.21	0.20	0.19	0.18	0.25	0.24
3.0	0.30	0.27	0.27	0.27	0.26	0.26	0.25	0.25	0.25	0.24	0.23	0.28	0.28
1.0	0.43	0.43	0.40	0.43	0.40	0.40	0.40	0.40	0.38	0.39	0.38	0.38	0.38
0.5	0.52	0.60	0.51	0.59	0.51	0.55	0.55	0.53	0.49	0.54	0.51	0.46	0.46
Avg	0.34	0.33	0.31	0.33	0.31	0.32	0.32	0.31	0.29	0.30	0.29	0.32	0.32

Table 16: Target marginal estimation ℓ_1 error with binning target psuedolabels across different DA methods aggregated grouped by shift severity in target label marginal.

Shift	Source	BN-adapt		TENT		DANN		CDANN		FixMatch		NoisyStudent	
	None	None	IS	None	IS	None	IS	None	IS	None	IS	None	IS
100.0	0.37	0.28	0.30	0.28	0.29	0.23	0.23	0.22	0.21	0.28	0.27	0.30	0.30
10.0	0.39	0.31	0.33	0.31	0.32	0.26	0.26	0.26	0.24	0.29	0.29	0.32	0.32
3.0	0.42	0.36	0.37	0.35	0.36	0.29	0.29	0.30	0.28	0.32	0.31	0.35	0.35
1.0	0.44	0.39	0.38	0.40	0.38	0.36	0.35	0.35	0.32	0.37	0.37	0.37	0.37
0.5	0.41	0.41	0.36	0.42	0.36	0.41	0.42	0.36	0.35	0.40	0.39	0.38	0.36
Avg	0.40	0.35	0.35	0.35	0.34	0.31	0.31	0.28	0.28	0.34	0.33	0.35	0.34

Table 17: Target marginal estimation ℓ_1 error with MLLS across different DA methods aggregated grouped by shift severity in target label marginal.

N Results on each dataset with source validation performance as early stopping criterion

In this section, we present results across all datasets. Different rows show different algorithm and $\{\text{None}, \text{RLLS}, \text{True}\}$ denote the re-weighting estimate used. 'None' implies no re-weighting of the classifier. Since IW-CDAN and IW-DAN already incorporate an estimate of target label marginal in their training procedure, we do not adjust the obtained classifier further with our re-weighting correction.

	Alpha = None			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
Algorithm	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	87.6	87.3	87.6	87.2	86.9	87.5	87.3	87.2	88.0	88.1	90.4	91.1	91.0	93.6	94.0
Source (w aug)	89.9	89.7	89.9	90.0	89.5	89.8	90.1	90.0	90.4	90.6	92.4	92.7	92.9	95.0	95.3
Source (adv)	59.8	33.2	59.8	61.8	34.8	55.1	64.2	38.1	54.0	56.1	42.2	67.4	54.9	51.9	68.7
Source (clip)	89.3	88.5	89.3	89.1	88.4	89.5	89.1	88.7	90.0	89.2	88.9	90.0	88.8	88.9	91.5
DARE	85.0	84.9	85.0	83.1	83.4	83.4	80.5	80.9	81.2	72.4	73.1	74.4	61.1	62.8	70.7
BN-adapt	90.4	90.2	90.4	89.3	89.5	89.7	87.9	89.5	89.9	85.4	89.8	91.1	80.3	89.9	93.1
RS-BN-adapt	90.5	90.3	90.5	90.3	90.2	90.6	90.5	90.5	91.1	91.1	92.8	93.0	91.2	95.4	96.0
TENT	90.7	90.4	90.7	89.4	89.6	89.9	87.9	89.5	89.8	85.6	90.1	91.3	80.1	90.1	93.1
RS-TENT	90.5	90.4	90.5	90.3	90.2	90.6	90.4	90.4	91.0	91.1	92.8	93.0	91.2	95.4	95.9
DANN	86.6	86.0	86.6	86.3	86.0	86.6	86.0	86.2	86.6	86.0	89.5	90.3	90.0	93.3	93.8
IW-DANN	86.6			86.4			86.2			85.9			89.8		
RS-DANN	85.1	83.2	85.1	84.7	83.0	84.6	84.3	83.1	84.3	85.2	87.0	88.2	88.9	91.4	91.7
CDANN	86.5	86.0	86.5	86.0	85.6	86.3	85.8	85.7	86.4	85.8	89.7	90.5	90.0	93.3	93.6
IW-CDANN	86.5			86.0			85.8			85.9			90.0		
RS-CDANN	86.6	86.0	86.6	86.4	85.8	86.6	85.7	85.5	86.4	86.6	90.2	90.5	90.3	93.3	93.7
FixMatch	91.0	90.9	91.0	91.2	91.2	91.3	91.3	91.4	91.8	91.1	93.4	93.5	91.3	95.1	95.8
RS-FixMatch	91.4	91.2	91.4	91.6	91.4	91.6	91.5	91.8	91.9	92.0	93.4	93.6	94.2	95.7	95.9
NoisyStudent	91.0	90.8	91.0	91.1	90.5	90.8	91.1	91.0	91.2	92.6	93.8	93.9	95.0	95.6	95.7
RS-NoisyStudent	91.0	90.8	91.0	91.1	90.5	90.8	91.1	91.0	91.2	92.6	93.8	93.9	95.0	95.6	95.7
SENTRY	88.6	88.3	88.6	88.4	88.4	88.7	88.4	88.6	88.9	88.8	91.3	91.9	89.1	93.9	94.4

Table 18: CIFAR10 results aggregated across different distribution shift pairs

	Alpha = None			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
Algorithm	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	64.1	58.0	64.1	65.6	59.1	65.3	64.7	58.1	66.0	64.4	62.1	72.6	66.6	69.3	79.2
Source (w aug)	70.2	66.7	70.2	71.5	67.3	71.8	70.2	65.7	72.3	69.8	70.2	77.2	71.5	76.2	82.4
Source (adv)	26.1	18.6	26.1	26.2	19.5	27.3	25.5	19.2	28.6	25.0	19.7	34.8	28.2	28.5	43.0
Source (clip)	83.0	82.6	83.0	84.2	83.8	84.4	84.4	84.1	85.0	83.9	86.1	88.0	85.2	87.1	91.6
DARE	64.1	63.9	64.1	63.4	63.2	63.7	59.0	58.1	59.4	46.5	46.5	53.6	37.4	38.6	58.3
BN-adapt	71.6	69.4	71.6	72.9	69.8	73.0	71.0	67.9	73.2	70.5	73.1	78.4	71.6	78.1	82.4
RS-BN-adapt	71.5	69.2	71.5	72.6	69.7	72.9	71.3	68.1	73.1	71.1	72.8	78.1	72.9	78.3	82.5
TENT	71.4	69.4	71.4	72.6	69.8	73.0	71.0	68.0	73.0	70.4	72.9	77.9	71.9	78.0	82.5
RS-TENT	71.5	69.2	71.5	72.4	69.5	72.7	71.3	68.2	73.0	71.4	72.7	78.1	73.0	78.3	82.5
DANN	78.3	77.8	78.3	78.7	78.5	79.0	77.9	77.4	79.0	75.7	79.9	82.0	78.7	83.4	86.1
IW-DANN	78.3			79.0			77.7			77.4			80.0		
RS-DANN	78.3	77.7	78.3	78.3	77.4	78.4	76.8	76.2	78.2	75.4	78.5	81.1	74.3	77.8	81.8
CDANN	77.9	77.3	77.9	77.9	77.6	78.3	75.9	75.6	77.0	75.7	78.2	80.5	79.4	82.3	86.4
IW-CDANN	77.7			77.6			76.4			77.1			79.3		
RS-CDANN	77.4	76.9	77.4	78.0	77.4	78.1	77.0	76.3	77.8	76.4	78.4	81.1	77.2	80.2	85.2
FixMatch	72.1	69.5	72.1	72.1	69.2	72.5	70.7	67.8	72.8	71.1	71.9	77.6	74.1	77.9	81.4
RS-FixMatch	71.9	69.6	71.9	72.6	70.2	73.3	71.1	68.4	73.0	71.6	73.1	78.1	73.6	77.1	81.6
NoisyStudent	71.3	68.4	71.3	71.7	68.8	72.3	70.4	66.9	71.3	72.1	72.8	77.0	73.9	78.0	80.8
RS-NoisyStudent	71.3	68.3	71.3	71.6	68.9	72.3	70.3	66.7	71.0	72.4	73.0	77.2	73.8	78.0	80.9
SENTRY	67.6	64.0	67.6	68.3	64.7	68.9	67.3	63.0	69.2	68.9	70.0	74.9	69.6	73.0	78.6

Table 19: CIFAR100 results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	81.8	81.8	81.9	82.2	82.3	82.3	82.4	82.6	82.6	79.6	81.7	85.0	79.8	81.4	84.0
Source (w aug)	78.0	77.1	78.0	79.4	78.7	79.3	80.5	80.0	80.5	68.6	67.4	74.3	69.5	68.4	74.0
Source (adv)	82.8	82.1	82.8	83.4	82.8	83.2	84.0	83.5	83.8	77.8	79.0	85.2	78.3	79.2	84.2
Source (clip)	96.1	96.1	96.1	96.3	96.3	96.3	96.5	96.4	96.4	94.8	95.3	95.5	94.9	95.3	95.5
DARE	79.0	79.0	79.1	79.5	79.5	79.5	79.7	79.8	79.9	73.9	74.2	76.9	74.4	74.8	76.9
BN-adapt	89.5	88.7	89.4	89.0	88.6	89.2	85.8	85.2	86.5	84.3	90.1	88.7	84.6	87.8	89.1
RS-BN-adapt	88.6	87.3	88.6	88.8	88.2	89.1	87.6	87.0	88.3	89.0	90.3	92.1	90.0	87.6	92.4
TENT	92.3	92.1	92.3	90.3	90.4	90.5	91.0	90.9	91.3	81.9	85.6	85.0	81.1	83.4	84.5
RS-TENT	92.0	90.4	92.0	92.7	92.7	92.9	90.9	90.9	91.2	85.1	90.3	89.6	86.6	87.6	89.7
DANN	83.0	82.7	83.1	84.2	83.9	84.2	85.2	85.1	85.3	75.2	74.6	79.0	78.2	78.3	80.7
IW-DANN	84.1			84.2			85.9			78.7			78.1		
RS-DANN	84.1	83.7	84.1	83.7	82.9	83.5	86.5	86.2	86.5	71.6	70.8	75.0	75.9	75.7	78.4
CDANN	87.3	87.1	87.3	87.3	87.0	87.4	87.0	87.0	87.1	80.7	81.4	86.8	79.7	79.8	83.3
IW-CDANN	87.2			85.3			85.5			81.6			86.1		
RS-CDANN	88.1	87.8	88.1	83.2	83.2	83.3	85.8	85.8	86.0	88.7	90.5	91.3	92.2	93.3	93.4
FixMatch	91.3	91.3	91.3	92.7	92.5	92.5	93.6	93.8	93.7	79.9	82.1	83.8	81.5	82.7	84.0
RS-FixMatch	88.6	87.8	88.6	93.7	93.5	93.6	94.2	94.2	94.2	81.5	82.9	84.2	80.1	80.6	82.8
NoisyStudent	88.4	88.1	88.4	85.9	84.9	85.9	88.2	87.9	88.3	83.0	83.7	86.1	84.4	85.4	86.8
RS-NoisyStudent	87.7	87.1	87.7	85.7	84.7	85.6	88.6	88.2	88.8	84.9	85.9	87.6	85.3	86.1	87.5
SENTRY	90.7	90.4	90.7	91.5	91.2	91.4	90.6	90.7	90.7	80.7	81.9	82.4	83.5	84.9	85.0

Table 20: Camelyon results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	73.4	73.1	73.4	74.3	74.2	74.7	76.8	77.5	78.4	80.0	81.1	83.8	78.6	80.3	83.8
Source (w aug)	78.2	78.3	78.2	80.2	79.9	80.1	82.3	82.5	83.1	85.3	86.3	87.1	81.5	85.0	86.1
Source (adv)	73.2	71.5	73.2	75.5	74.5	75.5	78.3	76.6	78.2	79.7	80.8	83.2	76.9	81.8	83.4
Source (clip)	88.8	88.9	88.8	89.8	90.1	90.1	90.3	90.5	90.6	92.4	92.7	94.4	91.5	92.4	94.1
DARE	73.9	74.0	73.9	73.5	73.5	73.9	72.4	72.1	72.7	63.9	64.3	72.8	55.5	57.2	64.7
BN-adapt	77.9	77.8	77.9	79.8	79.7	79.7	81.3	81.7	82.4	82.4	84.4	85.9	76.1	80.0	83.2
RS-BN-adapt	77.5	77.4	77.5	79.8	79.6	79.6	82.0	82.4	82.5	84.9	86.1	86.7	80.6	84.0	86.3
TENT	77.9	77.8	77.9	79.9	79.7	79.8	81.4	81.8	82.5	82.5	84.5	86.0	76.2	80.1	83.2
RS-TENT	77.5	77.5	77.5	79.9	79.6	79.7	82.1	82.4	82.5	84.9	86.1	86.7	80.6	84.0	86.3
DANN	75.7	75.3	75.7	77.4	77.5	77.9	79.1	80.2	80.6	80.7	83.4	84.7	79.3	80.9	82.1
IW-DANN	76.0			77.5			79.8			83.5			77.9		
RS-DANN	75.5	75.3	75.5	77.1	77.8	77.9	79.8	80.5	80.5	82.9	85.1	85.7	77.9	80.4	81.9
CDANN	76.2	75.9	76.2	77.5	77.8	78.8	78.7	79.7	80.5	82.4	86.0	87.2	77.7	81.6	83.3
IW-CDANN	75.4			77.2			78.5			83.3			79.2		
RS-CDANN	74.1	73.6	74.1	77.8	78.3	78.2	79.7	80.1	80.7	81.4	85.1	86.3	73.8	77.1	78.8
FixMatch	79.8	79.9	79.8	80.4	80.4	80.9	81.9	82.4	82.9	82.2	85.5	88.2	76.6	79.6	81.2
RS-FixMatch	80.5	80.4	80.5	81.0	81.4	81.6	82.6	83.1	83.8	87.5	88.1	89.2	79.9	83.5	85.4
NoisyStudent	78.9	78.5	78.9	78.4	78.5	78.8	80.9	81.5	81.8	85.2	86.5	88.2	82.8	85.7	88.0
RS-NoisyStudent	79.0	78.8	79.0	80.7	80.9	80.8	80.3	81.2	81.9	84.7	87.8	89.0	83.2	85.2	86.6
SENTRY	76.6	76.4	76.6	76.3	74.8	76.8	69.9	67.7	70.6	80.7	82.0	83.3	56.5	57.6	60.9

Table 21: Entity13 results aggregated across different distribution shift pairs

Algorithm	Alpha = None			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	63.5	63.1	63.5	62.9	63.4	63.6	63.4	64.6	65.1	64.4	67.9	74.7	56.0	63.5	78.4
Source (w aug)	70.2	69.8	70.2	71.2	70.5	70.9	70.6	70.1	71.4	72.4	73.8	75.7	64.8	70.5	81.3
Source (adv)	61.2	59.7	61.2	61.6	60.8	62.9	62.3	61.5	65.2	64.3	66.5	70.3	55.3	59.3	77.0
Source (clip)	85.3	85.0	85.3	85.8	85.3	85.8	85.4	85.4	85.9	85.8	87.9	89.2	81.3	83.5	89.2
DARE	66.7	66.4	66.7	65.3	65.7	65.2	62.2	62.4	63.4	50.8	50.4	57.5	27.3	28.4	53.6
BN-adapt	69.9	69.6	69.9	70.2	70.0	70.6	69.4	70.0	71.0	71.2	73.1	74.7	61.6	67.5	80.0
RS-BN-adapt	70.0	69.6	70.0	70.8	70.7	71.3	69.7	70.8	71.8	72.1	73.8	75.3	63.9	69.8	80.8
TENT	69.9	69.7	69.9	70.3	70.0	70.6	69.4	70.0	71.0	71.2	73.1	74.7	61.7	67.5	80.0
RS-TENT	70.1	69.7	70.1	70.8	70.5	71.5	69.8	70.8	71.9	72.1	73.8	75.3	63.9	69.8	80.8
DANN	67.3	67.0	67.3	67.5	67.1	67.5	66.2	66.6	68.8	66.9	68.6	75.3	61.0	65.4	80.3
IW-DANN	66.1			66.3			67.1			70.6			60.2		
RS-DANN	66.5	66.5	66.5	67.8	67.7	68.7	67.0	66.4	68.8	70.2	71.9	77.1	55.3	62.2	79.4
CDANN	65.8	66.1	65.8	66.8	65.9	66.6	66.8	66.1	68.9	69.0	70.9	74.9	55.4	61.6	76.8
IW-CDANN	67.8			66.2			65.7			70.0			53.4		
RS-CDANN	66.0	66.1	66.0	68.4	68.2	67.4	67.5	69.1	70.2	69.7	72.4	76.9	61.5	66.9	78.1
FixMatch	72.3	71.6	72.3	72.9	73.2	73.5	73.9	74.1	74.4	74.4	76.1	78.3	64.0	68.5	78.5
RS-FixMatch	73.4	72.7	73.4	72.2	71.5	71.9	71.3	72.4	73.6	70.4	72.9	77.5	60.4	68.4	79.4
NoisyStudent	70.5	69.4	70.5	70.0	69.8	70.5	71.7	71.0	72.2	72.9	74.2	76.7	63.6	65.8	76.0
RS-NoisyStudent	70.3	70.2	70.3	71.6	71.4	71.5	71.4	71.2	71.3	72.7	74.3	77.7	60.9	66.5	81.2
SENTRY	64.4	62.5	64.4	64.8	63.8	64.4	62.3	61.4	63.2	54.1	55.2	60.4	39.3	41.6	71.4

Table 22: Entity30 results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	66.3	65.4	66.3	68.9	68.8	68.9	69.8	69.8	72.3	66.0	68.9	76.9	64.8	66.9	88.6
Source (w aug)	72.8	71.6	72.8	75.8	74.2	75.9	76.1	74.0	75.5	71.5	72.5	81.1	76.3	78.9	86.5
Source (adv)	51.7	45.3	51.7	52.5	47.7	55.1	50.8	48.1	57.3	50.6	50.0	66.3	40.8	57.8	80.8
Source (clip)	90.0	89.9	90.0	92.9	92.8	93.0	93.9	93.9	94.2	91.9	92.0	94.7	96.4	97.6	98.0
DARE	66.6	66.2	66.6	66.2	66.2	66.3	59.8	58.9	61.3	43.8	44.2	57.0	29.1	31.4	54.8
BN-adapt	73.4	72.7	73.4	75.4	74.0	75.5	74.2	73.4	74.0	69.3	70.7	79.5	65.6	69.5	85.0
RS-BN-adapt	72.4	72.0	72.4	75.2	73.8	75.4	74.5	74.6	73.9	70.7	72.2	80.6	62.8	72.1	86.3
TENT	72.2	71.8	72.2	74.4	73.7	74.6	74.3	73.5	74.0	69.4	70.9	79.5	65.6	69.3	85.0
RS-TENT	72.4	72.0	72.4	75.2	73.8	75.4	74.5	74.6	73.9	70.7	72.2	80.6	62.8	72.1	86.3
DANN	69.7	68.2	69.7	72.1	71.4	73.1	73.1	73.3	74.8	72.8	73.7	79.9	54.8	70.1	85.5
IW-DANN	68.6			74.4			72.9			72.2			71.9		
RS-DANN	68.7	67.9	68.7	70.0	70.0	70.3	72.3	71.4	71.8	74.6	77.8	81.8	66.9	70.6	84.7
CDANN	68.5	68.0	68.5	70.2	70.1	71.1	72.7	71.5	75.2	72.3	73.4	77.3	67.4	76.5	91.7
IW-CDANN	70.7			71.5			76.2			70.2			61.0		
RS-CDANN	71.6	70.5	71.6	73.6	72.9	73.8	71.4	69.4	74.1	72.0	72.7	83.4	67.5	76.8	85.1
FixMatch	76.2	76.3	76.2	77.0	76.3	77.8	78.2	78.2	80.5	78.1	79.1	87.0	66.0	69.3	84.9
RS-FixMatch	74.2	74.2	74.2	78.4	78.3	79.0	78.2	78.2	79.5	80.2	83.2	86.2	67.9	70.6	86.3
NoisyStudent	70.5	71.4	70.5	77.1	76.4	77.3	76.5	74.3	75.9	77.4	75.4	83.4	71.5	73.8	87.4
RS-NoisyStudent	73.0	71.6	73.0	77.3	77.1	77.3	75.2	76.1	77.6	70.5	75.2	87.9	58.9	79.2	87.2
SENTRY	65.5	64.4	65.5	65.4	62.6	65.6	48.9	49.0	48.1	48.3	48.8	63.2	43.5	40.4	75.9

Table 23: Living17 results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE None	RLLS	True	Alpha = 10.0 None	RLLS	True	Alpha = 3.0 None	RLLS	True	Alpha = 1.0 None	RLLS	True	Alpha = 0.5 None	RLLS	True
Source (w/o aug)	54.1	54.0	54.1	53.2	52.5	52.7	54.3	53.6	54.7	62.5	62.8	67.2	49.5	51.6	65.5
Source (w aug)	62.7	61.9	62.7	62.6	61.7	62.2	60.9	60.3	63.7	62.3	65.9	71.5	59.0	64.1	72.4
Source (adv)	55.8	53.2	55.8	54.0	50.2	55.1	56.2	53.0	56.7	58.4	61.2	66.7	46.5	51.7	61.2
Source (clip)	82.4	82.5	82.4	84.4	84.4	84.5	86.6	85.9	88.4	83.3	81.9	88.6	85.7	86.3	91.6
DARE	55.8	55.3	55.8	57.2	57.6	57.2	51.7	52.7	54.6	41.9	42.2	60.9	19.4	20.6	44.4
BN-adapt	62.4	62.1	62.4	62.0	61.2	61.9	60.0	59.8	63.2	61.0	65.1	71.0	55.9	62.2	71.4
RS-BN-adapt	62.5	62.1	62.5	62.7	61.3	62.4	60.3	59.7	63.2	63.2	65.7	69.8	60.7	63.0	72.1
TENT	62.6	62.3	62.6	62.0	61.2	61.9	60.0	59.8	63.2	61.0	65.1	71.0	55.9	62.2	71.4
RS-TENT	62.5	62.2	62.5	62.7	61.4	62.4	60.3	59.8	63.2	63.2	65.7	69.8	60.7	63.0	72.1
DANN	59.6	59.0	59.5	59.2	57.2	58.4	55.6	57.6	57.8	62.4	64.3	70.3	59.7	65.4	74.2
IW-DANN	59.6			61.2			60.3			62.3			56.4		
RS-DANN	57.9	57.5	57.9	56.1	55.5	56.2	58.6	57.8	58.2	57.3	57.6	69.8	53.6	54.4	73.9
CDANN	58.0	57.2	58.0	57.4	57.0	58.1	59.4	58.8	61.6	55.8	60.2	67.6	50.6	56.9	70.7
IW-CDANN	59.6			57.5			57.8			61.8			56.6		
RS-CDANN	59.9	58.4	59.9	57.5	58.0	57.6	59.4	61.4	60.9	59.9	59.7	68.7	56.9	62.6	74.2
FixMatch	66.2	65.4	66.2	64.8	62.9	66.4	62.4	63.6	66.1	64.9	64.1	75.7	52.7	53.3	81.3
RS-FixMatch	64.6	62.8	64.6	67.5	66.9	67.5	64.7	64.0	64.1	61.3	61.8	72.0	56.2	61.3	73.9
NoisyStudent	64.2	63.1	64.1	61.8	63.0	62.7	60.0	59.4	61.9	66.4	67.0	72.3	57.1	59.1	75.5
RS-NoisyStudent	62.1	61.9	62.1	63.2	61.3	63.4	59.9	60.7	62.6	65.9	67.6	74.8	62.2	64.8	75.8
SENTRY	55.9	54.1	55.9	41.2	35.5	42.5	46.4	42.1	48.5	39.5	38.7	49.0	24.5	24.4	58.7

Table 24: Nonliving26 results aggregated across different distribution shift pairs

Algorithm	Alpha = None			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	56.4	55.7	56.5	57.7	57.2	57.8	57.6	58.2	58.7	59.7	62.1	64.1	55.7	59.1	61.9
Source (w aug)	59.7	57.9	59.8	60.5	58.9	60.9	60.1	59.9	62.1	61.7	64.5	67.9	58.6	63.3	67.5
Source (adv)	48.4	44.0	48.6	49.3	45.2	49.8	49.7	45.7	51.4	51.5	50.2	57.9	48.7	50.6	57.0
Source (clip)	64.0	63.2	64.1	65.1	64.2	65.4	65.3	65.3	66.9	66.3	68.2	71.6	61.8	66.4	69.9
DARE	53.9	53.6	54.0	54.0	54.1	54.4	51.7	51.5	52.6	45.7	46.7	52.1	41.5	41.8	50.3
BN-adapt	57.1	55.7	57.3	57.6	56.3	58.0	56.9	56.1	58.8	58.5	60.7	64.8	53.7	58.7	62.9
RS-BN-adapt	56.8	55.5	57.0	57.2	55.3	57.7	56.7	55.5	58.5	60.0	60.9	65.2	55.0	59.0	64.0
TENT	57.2	54.5	57.4	58.7	57.5	59.0	58.9	57.7	60.4	59.6	61.0	64.4	55.8	60.2	63.9
RS-TENT	58.0	55.0	58.1	59.1	57.7	59.6	57.6	56.0	59.2	59.5	61.3	64.6	55.0	59.0	63.7
DANN	57.2	55.8	57.3	59.1	57.7	59.1	57.8	56.8	59.2	59.5	60.2	64.6	55.3	59.3	63.0
IW-DANN	57.1			58.7			58.1			56.3			55.2		
RS-DANN	56.6	55.1	56.8	57.2	55.6	57.6	57.2	56.5	58.9	59.2	59.4	63.0	53.9	57.9	60.9
CDANN	57.3	55.9	57.4	58.3	56.7	58.5	58.2	56.7	59.8	58.2	59.0	63.8	54.9	57.5	62.4
IW-CDANN	57.2			58.1			57.9			59.3			53.3		
RS-CDANN	56.3	55.4	56.5	57.3	56.6	57.7	56.2	54.8	58.0	56.3	57.5	61.3	54.6	56.8	60.6
FixMatch	59.6	58.3	59.6	60.8	59.3	60.9	61.3	60.4	62.5	62.0	63.6	66.6	58.1	62.2	65.5
RS-FixMatch	57.6	56.2	57.5	58.3	57.5	58.7	57.8	57.0	59.3	59.0	59.2	64.4	54.7	59.0	63.2
NoisyStudent	61.5	60.3	61.5	62.1	60.5	62.1	60.5	60.3	61.7	61.8	62.5	66.0	57.2	61.5	64.9
RS-NoisyStudent	61.2	59.7	61.1	61.7	60.0	61.8	61.3	60.6	62.3	61.2	62.0	66.4	59.5	63.3	66.3
SENTRY	51.4	46.0	51.3	51.7	47.4	52.0	50.0	45.5	51.7	50.0	46.6	55.9	45.0	44.1	55.6

Table 25: FMoW results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	49.4	45.1	51.0	49.4	45.5	51.8	49.0	45.7	52.5	50.1	47.7	56.5	48.6	48.4	59.0
Source (w aug)	53.0	49.2	54.6	53.0	49.6	55.2	52.5	49.7	56.1	53.1	51.6	60.4	52.8	52.8	61.9
Source (adv)	48.7	43.5	50.5	48.8	43.6	51.2	48.6	44.2	52.3	49.5	46.8	56.0	49.0	47.3	58.4
Source (clip)	70.8	68.8	72.2	71.4	69.4	73.6	71.2	69.5	73.9	71.4	70.9	76.3	71.7	72.0	79.1
DARE	51.5	51.1	53.0	50.4	49.9	52.6	48.3	48.0	52.2	45.8	45.7	53.8	38.3	39.3	51.5
BN-adapt	53.7	52.7	55.6	53.6	52.5	56.6	53.3	52.7	57.4	54.4	54.2	61.4	52.2	54.7	62.5
RS-BN-adapt	54.1	52.7	55.9	53.9	52.9	56.6	53.2	52.6	57.2	54.5	54.1	62.1	52.4	54.5	62.6
TENT	55.3	54.4	57.0	54.2	53.3	57.2	54.0	53.5	58.2	54.5	54.6	61.5	52.4	55.2	62.8
RS-TENT	55.8	55.1	57.6	54.7	53.7	57.3	53.9	53.2	58.1	54.6	54.2	61.9	52.9	55.0	63.0
DANN	53.8	53.1	55.8	53.5	53.0	56.7	52.1	51.6	56.3	52.4	52.2	59.4	47.3	49.3	59.6
IW-DANN	54.2			53.7			52.7			51.3			48.4		
RS-DANN	55.1	54.3	56.4	54.9	54.3	57.2	54.1	53.4	56.9	52.2	52.2	59.0	51.0	52.0	60.1
CDANN	55.7	54.8	57.1	54.6	53.5	57.2	54.1	53.1	57.2	53.4	53.0	60.4	52.2	54.2	60.5
IW-CDANN	55.9			55.0			54.4			53.8			51.6		
RS-CDANN	56.2	55.3	57.5	55.5	54.4	57.8	54.7	53.9	57.7	54.1	53.8	60.7	53.4	54.0	62.0
FixMatch	57.6	55.7	59.0	57.6	55.8	59.6	57.6	55.8	60.0	58.4	57.4	63.1	58.4	58.8	63.8
RS-FixMatch	58.6	56.6	59.1	58.0	56.2	59.4	57.7	56.0	59.5	59.1	58.3	63.2	58.6	58.8	64.4
NoisyStudent	54.9	52.0	56.1	53.7	50.8	55.8	53.5	51.1	56.2	54.1	52.8	60.2	55.5	55.2	63.1
RS-NoisyStudent	54.6	51.3	55.8	53.5	50.6	55.3	53.6	50.7	56.3	54.9	52.9	60.4	54.9	54.9	62.7
SENTRY	57.3	56.0	57.4	49.0	45.2	50.1	51.6	49.0	53.6	48.3	46.0	53.7	46.3	44.9	53.7

Table 26: DomainNet results aggregated across different distribution shift pairs

Algorithm	Alpha = NONE			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	62.6	58.5	62.7	63.7	60.3	64.3	65.0	61.1	65.8	66.4	67.2	73.1	62.0	63.0	73.4
Source (w aug)	62.2	59.2	62.0	63.3	59.7	64.5	64.5	62.4	66.5	66.5	68.9	72.1	66.4	66.4	73.4
Source (adv)	56.5	50.9	56.3	58.5	52.6	59.6	59.5	55.5	60.7	60.7	61.3	70.5	60.1	54.5	72.1
Source (clip)	79.8	77.3	80.2	79.2	77.5	79.9	78.9	77.5	80.2	79.4	78.7	83.9	78.5	79.3	88.0
DARE	59.4	59.2	60.6	56.5	56.4	59.2	53.1	52.4	58.1	39.6	37.9	56.7	32.6	32.9	59.7
BN-adapt	62.8	61.3	64.9	65.0	64.1	66.4	66.4	64.8	69.1	67.0	69.2	75.5	67.2	68.2	76.2
RS-BN-adapt	63.2	60.9	64.1	65.3	62.8	66.0	66.9	63.3	68.7	67.6	68.7	74.3	66.7	67.7	76.2
TENT	62.8	61.6	64.8	65.1	63.7	66.0	66.0	65.3	68.8	67.3	69.2	74.9	66.7	68.9	75.4
RS-TENT	63.2	61.0	64.1	65.1	62.9	66.2	66.9	63.4	68.8	67.3	69.4	74.3	66.7	67.9	76.2
DANN	66.6	65.1	67.5	67.2	65.6	67.8	67.7	67.7	69.4	68.3	70.2	77.2	62.9	64.7	75.2
IW-DANN	66.9			67.6			67.7			67.7			63.1		
RS-DANN	67.5	65.3	68.0	67.1	65.4	68.0	67.6	65.0	69.4	70.7	70.1	76.6	65.4	67.4	75.4
CDANN	66.3	65.0	66.9	66.5	65.8	67.7	66.9	65.6	68.7	68.7	70.8	77.9	63.9	65.7	75.5
IW-CDANN	66.3			66.6			66.7			68.6			63.4		
RS-CDANN	65.2	63.5	65.9	66.9	65.1	66.8	67.6	65.1	67.0	65.7	66.3	75.1	63.2	61.0	75.2
FixMatch	62.5	57.9	62.5	64.1	58.7	63.9	65.9	60.9	65.1	65.2	66.9	72.9	66.2	67.4	75.9
RS-FixMatch	62.9	57.1	61.5	63.4	57.8	62.9	65.3	58.3	64.8	68.9	66.4	72.7	62.4	63.4	72.1
NoisyStudent	65.0	61.3	65.2	65.6	63.1	66.4	67.0	65.5	68.1	69.9	68.0	75.6	66.3	65.6	76.7
RS-NoisyStudent	65.3	62.0	64.9	65.4	62.8	66.1	67.3	65.3	67.6	70.4	68.4	75.8	65.8	65.8	76.2
SENTRY	58.1	53.1	57.9	58.4	52.7	59.6	59.8	55.4	61.1	62.1	61.3	68.8	54.1	53.8	68.0

Table 27: Officehome results aggregated across different distribution shift pairs

Algorithm	Alpha = None			Alpha = 10.0			Alpha = 3.0			Alpha = 1.0			Alpha = 0.5		
	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True	None	RLLS	True
Source (w/o aug)	64.0	60.4	63.7	62.8	59.6	63.6	61.7	58.9	63.9	57.5	56.2	62.4	65.8	67.6	75.8
Source (w aug)	60.8	57.9	60.6	59.2	56.5	59.8	57.7	55.3	59.2	55.7	54.1	58.0	65.4	66.3	73.3
Source (adv)	57.3	54.2	57.0	55.6	52.7	56.0	54.0	51.3	55.0	51.6	49.8	53.9	60.1	59.7	70.0
Source (clip)	75.5	73.2	75.5	74.8	72.8	75.7	74.2	72.5	76.1	73.1	72.7	76.7	78.2	79.6	82.1
DARE	63.3	63.2	63.7	61.7	61.7	62.7	60.3	60.4	62.0	57.4	57.8	60.7	52.0	54.9	63.5
BN-adapt	71.5	71.3	72.3	71.4	71.4	73.2	69.9	70.1	73.0	65.9	67.3	73.3	57.1	63.2	74.5
RS-BN-adapt	70.1	68.8	70.4	69.3	68.5	70.6	67.6	67.0	69.8	64.1	63.5	68.2	67.8	71.0	76.8
TENT	73.6	73.6	74.5	72.5	72.5	74.2	70.9	71.3	74.0	66.6	67.4	73.4	58.5	64.5	74.7
RS-TENT	71.7	70.7	72.0	70.7	70.2	71.9	68.4	68.0	70.7	64.4	63.9	68.3	68.2	71.3	76.9
DANN	76.0	75.9	76.3	73.8	73.9	74.4	72.4	72.6	73.6	64.9	65.0	66.8	54.0	54.3	60.3
IW-DANN	77.0			73.5			71.7			64.0			53.6		
RS-DANN	77.5	77.4	77.6	76.8	76.8	77.3	77.0	77.3	78.3	68.9	68.8	70.8	59.3	60.2	66.5
CDANN	80.3	80.3	80.4	78.1	78.2	78.4	74.4	74.5	75.1	64.6	64.4	65.8	57.8	58.1	60.2
IW-CDANN	79.4			78.5			74.5			64.4			56.4		
RS-CDANN	79.7	79.6	79.7	76.7	76.6	76.9	79.8	79.9	80.4	71.4	71.0	72.4	64.1	64.4	66.4
FixMatch	80.7	80.7	81.4	77.5	77.3	78.7	75.9	75.8	78.5	73.1	73.6	78.8	60.3	63.5	71.9
RS-FixMatch	80.8	80.7	81.2	80.7	80.5	81.4	79.7	79.6	82.0	72.9	72.7	77.7	72.3	74.9	79.9
NoisyStudent	62.6	59.8	62.6	60.1	57.7	60.8	59.0	56.9	60.3	57.5	57.0	58.8	66.3	67.2	72.4
RS-NoisyStudent	62.3	59.7	62.4	60.6	58.0	61.1	58.4	56.2	60.0	57.1	56.1	58.1	67.8	68.6	73.9
SENTRY	78.3	77.9	78.6	81.3	81.0	81.9	79.8	79.0	81.4	72.2	72.7	73.9	74.5	76.8	80.8

Table 28: Visda results aggregated across different distribution shift pairs