MAX-MARGIN INSPIRED PER-SAMPLE RE-WEIGHTING FOR ROBUST DEEP LEARNING

Ramnath Kumar
Google AI Research Lab,
Bengaluru, India 560016,
ramnathk@google.com

Kushal Majmundar
Google AI Research Lab,
Bengaluru, India 560016,
majak@google.com

Dheeraj Nagaraj
Google AI Research Lab,
Bengaluru, India 560016,
dheerajnagaraj@google.com

Arun Suggala
Google AI Research Lab,
Bengaluru, India 560016,
arunss@google.com

ABSTRACT

We design simple, explicit, and flexible per-sample re-weighting schemes for learning deep neural networks in a variety of tasks that require robustness of some form. These tasks include classification with label imbalance, domain adaptation, and tabular representation learning. Our re-weighting schemes are simple and can be used in combination with any popular optimization algorithms such as SGD, Adam. Our techniques are inspired by max-margin learning, and rely on mirror maps such as log-barrier and negative entropy, which have been shown to perform max-margin classification. Empirically, we demonstrate the superiority of our approach on all of the aforementioned tasks. Our techniques provide state-of-the-art results in tasks involving tabular representation learning and domain adaptation.

1 INTRODUCTION

Deep learning has revolutionized artificial intelligence and machine learning with significant advancements in various domains such as image recognition Brock et al. (2021), natural language processing Schulman et al. (2022), drug discovery Vamathevan et al. (2019), and reinforcement learning Mnih et al. (2013). One of the most crucial aspects of their success is the suite of optimization algorithms used to train deep networks. In convex machine learning, optimization algorithms are deployed with pre-conditioning and per-sample weighting, boosting the convergence of learning algorithms. Our work aims to design simple, computationally efficient per-sample weighting schemes for deep learning by adapting update rules from mirror descent for binary classification tasks.

We begin by noting that the loss functions are usually designed as the population means of in-distribution samples. This can cause neural networks trained with SGD on such loss functions do not learn well on rare samples. This has been studied extensively in various domains, including computer vision, tabular data, meta-learning, and many more. Our goal is to minimize average losses, not individual sample losses. To address this, two related techniques have gained attention: pre-conditioning and per-sample re-weighting. These techniques improve model performance and make optimization more efficient. Pre-conditioning adjusts gradient step sizes in specific directions, with the Newton method being the most famous approach. Per-sample re-weighting adjusts the importance of each sample based on its loss value, with some approaches giving more weight to samples with higher loss and others giving more weight to samples with lower loss. This often involves training a second neural network to compute weights.

In this work, we propose MAX-MARGIN INSPIRED RE-WEIGHTED GRADIENT DESCENT (RGD). This modifies the gradient of the loss from $\sum_i \nabla l_i$ to $\sum_i g(l_i)\nabla l_i$ for some appropriately chosen

function scalar function $g()$. We derive a simple and exact form for $g()$ by drawing inspiration from mirror descent algorithms designed to obtain max-margin classifiers. In our experiments, we show that RGD outputs models with various robustness properties, including robustness to domain shifts, class imbalance, and label noise. Unlike many previous approaches, RGD does not require a separate neural network for re-weighting and thus has the same computational complexity as training without RGD. This allows each sample's weight to change dynamically based on its current loss while reducing the computational overhead.

Our approach is tested on the CIFAR dataset with class imbalance and label noise, demonstrating superior performance compared to specialized techniques like class-balanced loss and focal loss. Our approach also improves upon the state-of-the-art in two challenging tasks, DomainBed and tabular representation learning, by simply applying our method on top of the current state-of-the-art approach.

Domain generalization is a task where test data has a different distribution than training data. Max-margin type robust classifiers are expected to improve performance in this task. Gulrajani & Lopez-Paz (2020) showed that Empirical Risk Minimization applied over a deep network was highly effective and remained the state-of-the-art for a long time. Recent works such as Cha et al. (2022); Addepalli et al. (2022) showed improvement on this challenging task. Our method, RGD, improves upon the state-of-the-art results by being applied on top of the methods in Addepalli et al. (2022).

Learning with tabular data is a task where traditional machine learning methods, like random forests and GBDT, remain highly competitive against deep learning methods. Majmundar et al. (2022) recently introduced tabular representation learning methods, MET-S and MET, which improved upon the state-of-the-art performance of GBDT. Adversarial training, used in MET, indirectly promotes max-margins. Our re-weighting technique, RGD, provides a more direct and principled approach to learn max-margin classifiers. MET-S augmented with RGD significantly improves upon MET, providing a much more significant boost in performance.

Below, we present the main contributions of our paper:
A new per-sample weight formulation: We derive RGD by considering re-weighting schemes in general and deriving computationally efficient approximations for deep learning. We then connect our method to max-margin classification and mirror descent. We use this connection to choose our re-weighting scheme. This change is easy and efficient to implement with a few lines of code. Unlike many prior works, this does not require additional neural networks to learn per-sample weighting.
Flexible Weight Schemes: Our weighting schemes are flexible - some of the schemes improve the classification of rare data points by giving more weight to data points with a large loss while others improve robustness to label noise by giving more weight to samples with a smaller loss. This flexibility can be seen in our empirical studies on CIFAR classification with label imbalance (which reduces the occurrence of some classes, Section 3.1) and CIFAR classification with label noise (which adds noise to the labels, Section 3.2) respectively.
Better Performance Against SOTA: Our experimental results demonstrate that our proposed approach, RGD, significantly outperforms the state-of-the-art on various domains such as DomainBed (out-of-domain generalization), Tabular Representation Learning, Class Imbalance and Noisy Label domains with simple off-the-bat addition to the current SOTA techniques. As mentioned in the introduction, we show improvements on notoriously challenging problems and push the SOTA further on DomainBed and Tabular benchmarks by **+0.7%** and **+1.44%** respectively. Furthermore, we also show an average improvement of **+0.79%** on Class Imbalance experiments and **+2.33%** on Noisy Label experiments over their respective SOTA approach.(see Section 3)

In consideration of limited space, a comprehensive overview of related works can be found in Appendix 6.

2 Algorithm and Derivation

Our main algorithm is given in Algorithm 1. The main idea is that whenever a learning algorithm requests a mini-batch gradient, instead of sending $\frac{1}{B}\sum_i \nabla \ell_i$, we send the weighted stochastic gradient $\frac{1}{B}\sum_i g(\ell_i)\nabla \ell_i$ for some approriately chosen function $g : \mathbb{R}^+ \to \mathbb{R}$. In this work we consider the following choices for g: $g(x) = \frac{1}{1-\frac{\min(x,M)}{M+1}}$, $g(x) = \exp(\min(x, M))$, for some

Algorithm 1 Our Proposed Max-Margin inspired Re-weighted Gradient Descent (RGD)

1: **Input:** Data $\{X_i, Y_i\}_{i=1}^n$, learning rate η, number of iterations T, loss function ℓ, per-sample weighting function g, mini-batch size B
2: **for** $t = 1 \dots T$ **do**
3: Sample minibatch $\{X_i, Y_i\}_{i=1}^B$
4: Compute losses for points in the minibatch: $\ell_i \leftarrow \ell(X_i, Y_i; \theta),\ \forall i \in 1 \dots B$
5: Compute per-sample weights using our proposed approach: $w_i \leftarrow g(\ell_i)\ \forall i \in 1 \dots B$
6: Compute the weighted pseudo-gradient: $v \leftarrow \frac{1}{B} \sum_{i=1}^B w_i \nabla \ell_i$
7: Update weights of neural network: $\theta \leftarrow \theta - \eta v$
8: **end for**

constant $M > 0$. These functions are inspired by max-margin learning. We defer the derivation of these functions to Section 2.1. Observe that these functions give more weight to samples with high loss, which helps to learn with rare examples. However, this can be detrimental when the data is noisy. In such a scenario, we need to give more weight to low-loss samples to remove the noisy data points from hindering the learning process. In this scenario, we consider $g(x) = \exp(-\min(x, M))$ and $g(x) = (1 - \frac{\min(x,M)}{M+1})$.

For brevity and future references across the paper, we will denote our approach with $g(x) = \frac{1}{1-\frac{\min(x,M)}{M+1}}$ as RGD-1, and $g(x) = \exp(\min(x, M))$ as RGD-EXP. Their respective inverses (i.e, $g(x) = 1 - \frac{\min(x,M)}{M+1}$ and $g(x) = \exp(-\min(x, M))$) by *inv* RGD-1and *inv* RGD-EXP.

In the rest of this section, we present the framework of prioritized gradient descent and then derive our weighting function by appealing to the max-margin theory.

Suppose we are given $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ from a joint distribution $\mu(X, Y) = P(X)Q(Y|X)$ and our task is to fit Y to X by minimizing the parametric loss $\ell(X, Y; \theta)$, where θ describes the parameters of an ML model. We assume that $\theta \to \ell(X, Y; \theta)$ is sufficiently smooth (i.e., continuously differentiable). We minimize the expected population loss: $\mathcal{L}(\theta; P, Q) = \int \ell(X, Y; \theta) P(dX) Q(dY|X)$.

We will call the model well-specified if there exists θ^* such that $\theta^* \in \arg\min_\theta \int \ell(X, Y; \theta) Q(dY|X)$ for every X. This model is good for every X and robust to co-variate distribution shifts. To give a simple example, if the ground truth satisfies $Y = f(X, \theta^*) + \eta$ where η is a zero mean noise independent of X, this model is well-specified under the square loss $\ell(X, Y; \theta) = \mathbb{E}(Y - f(X, \theta))^2$.

Suppose we have a current estimate for θ; we consider changing the distribution of X from $P(\cdot)$ to a distribution dependent on θ, which prioritizes points where the learning has not progressed. Below we show that θ^* remains the risk minimizer under this new distribution.

Lemma 1. *Suppose the model is well specified as defined above. Then, given any class of measures (not necessarily probability distributions) $P(;\theta)$ over $\mathcal{X}$, parametrized by θ, we have: $\mathcal{L}(\theta^*; P(;\theta), Q) \leq \mathcal{L}(\theta; P(;\theta), Q)$.*

By Lemma 1, we argue that we can run GD-type methods by picking the data points X from the distribution $P(X; \theta)Q(Y|X)$ instead of $P(X)Q(Y|X)$ and still have θ^* to be a fixed point. More precisely, define Prioritized Gradient Descent (PGD) as:

$$\theta_{t+1} = \theta_t - \alpha \int P(dX; \theta_t) Q(dY|X) \nabla_\theta \ell(X, Y; \theta_t) \tag{1}$$

Lemma 2. *θ^* is a fixed point of equation 1.*

With empirical risk minimization, we consider labeled pairs $(X_i, Y_i)_{i=1}^n$ instead of the population. Given the prioritization distribution $P_t(X_i)$ such that $\sum_{i=1}^n P_t(X_i) = 1$, we write PGD for the empirical risk as:

$$\theta_{t+1} = \theta_t - \alpha \sum_{i=1}^n P_t(X_i) \nabla_\theta \ell(X_i, Y_i; \theta_t) \tag{2}$$

Since we want a mini-batch SGD-style algorithm which approximates equation 2, at each time t, draw a random batch B_t of size B uniformly at random from $\{X_1, \ldots, X_n\}$ and run:

$$\theta_{t+1} = \theta_t - \alpha \frac{n}{B} \sum_{i \in B_t} P_t(X_i) \nabla_\theta \ell(X_i, Y_i; \theta_t) \tag{3}$$

Note that this is more computationally efficient than sampling a batch size of B from the distribution $P_t(X_i)$ since the latter option has to parse the entire sample.

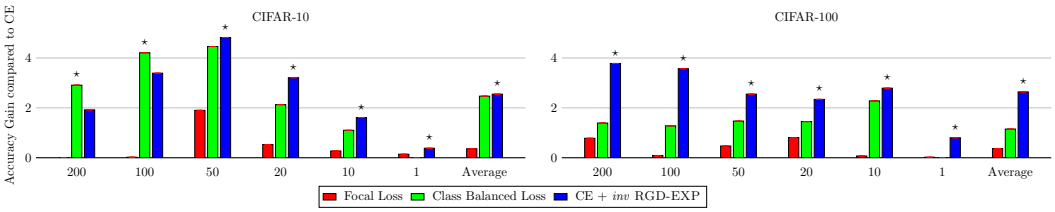

Figure 1: Ablation Study of RGD over the baseline Cross Entropy Loss (CE) on CIFAR-10 and CIFAR-100 datasets. We use the difference of accuracy (over the performance obtained when the baseline model is trained on Cross Entropy loss) to show that our proposed approach outperforms SOTA methods such as Focal Loss and Class Balanced Loss. $\star$ highlights the best-performing model in both settings. The x-axis refers to the Imbalance factor which is defined as the number of training samples in the largest class divided by the smallest. For more details refer to Table 11.1 in Appendix.

2.1 Max-margin inspired re-weighting

In this section, we design a weighting function inspired by max-margin learning. Consider the binary classification problem with linear separator, where we are given n samples $\{X_i, Y_i\}_{i=1}^n$. Here $X_i \in \mathbb{R}^d$ is the feature vector and $Y_i \in \{-1, 1\}$ is the label. We aim to find a linear classifier $\theta \in \Theta$ that best fits the data. The *minimum margin* of θ over the data is given by $\min_{i \in [n]} Y_i \langle \theta, X_i \rangle$. In max-margin learning, we aim to learn a classifier with the largest possible *minimum margin*. This leads us to the following objective: $\max_{\theta \in \Theta} \min_{i \in [n]} Y_i \langle \theta, X_i \rangle$.

Since the cross-entropy loss is a strictly monotonic function of its argument, one could replace $Y_i \langle \theta, X_i \rangle$ in the above objective with cross-entropy loss and obtain the following equivalent optimization problem: $\max_{\theta \in \Theta} \min_{i \in [n]} -\ell_{\text{CE}} (Y_i \langle \theta, X_i \rangle)$, where $\ell_{\text{CE}}(s) = \log(1 + e^{-s})$. The above problem can be rewritten as

$$\min_{\theta \in \Theta} \max_{P \in \Delta_n} \mathbb{E}_{i \sim P} \left[\ell_{\text{CE}} \left(Y_i \langle \theta, X_i \rangle \right) \right].$$

Note that the above optimization problem is convex in θ and concave in P. A popular and widely used approach for solving such problems is to rely on online learning algorithms (Hazan, 2016; Cesa-Bianchi & Lugosi, 2006). In this approach, the minimization player and the maximization player play a repeated game against each other. Both players rely on online learning algorithms to choose their actions in each round of the game, intending to minimize their respective regret. Such a procedure is known to converge to a Nash equilibrium of the game (see Remark 7.4 of Cesa-Bianchi & Lugosi, 2006). In this section, we use projected gradient descent for the minimization player and Online Mirror Descent (OMD) with regularizer $\mathcal{R}$ for the maximization player (Foster et al., 2016) (the updates for OMD with entropic regularizer can be found in the Appendix). Letting (θ_t, P_t) be the actions chosen by both the players in the t^{th} iteration of this repeated gameplay, we have

$$\theta_t \leftarrow \arg\min_{\theta \in \Theta} \left\langle \theta, \mathbb{E}_{i \sim P_{t-1}} \left[\nabla_\theta \ell_{\text{CE}}(Y_i \langle X_i, \theta_{t-1} \rangle) \right] \right\rangle + \frac{1}{2\eta} \|\theta - \theta_{t-1}\|_2^2,$$

$$P_t = \arg\max_{P \in \Delta_n} \sum_{s=1}^{t} \mathbb{E}_{i \sim P} \left[\ell_{\text{CE}}(Y_i \langle X_i, \theta_s \rangle) \right] + \frac{1}{\gamma} \mathcal{R}(P).$$

Here η, γ are the learning rates of θ, P.

Log-barrier regularizer: Picking $\mathcal{R}(P) = \sum_{i=1}^n \log P(i)$, we obtain the following closed-form

Table 1: Results on CIFAR-10 and CIFAR-100 dataset with flip noise.

Dataset	CIFAR-10				CIFAR-100			
Loss	0%	20%	40%	**Avg.**	0%	20%	40%	**Avg.**
Focal Loss	93.03 ± 0.16	86.45 ± 0.19	80.45 ± 0.97	86.64	70.02 ± 0.53	61.87 ± 0.30	54.13 ± 0.40	62.01
D2L	92.02 ± 0.14	87.66 ± 0.40	83.89 ± 0.46	87.86	68.11 ± 0.26	63.48 ± 0.53	51.83 ± 0.33	61.14
Cross Entropy (CE)								
Default	92.89 ± 0.32	76.83 ± 2.30	70.77 ± 2.31	80.16	70.50 ± 0.12	50.86 ± 0.27	43.01 ± 1.16	54.79
inv RGD-1 **(Ours)**	93.13 ± 0.17	90.91 ± 0.21	86.05 ± 0.28	90.03	71.31 ± 0.21	67.09 ± 0.23	54.06 ± 0.38	64.15
inv RGD-EXP **(Ours)**	**93.21** ± 0.26	**91.19** ± 0.14	**86.39** ± 0.61	**90.26**	**71.40** ± 0.18	**67.32** ± 0.24	**54.08** ± 0.77	**64.27**

expressions for the above updates

$$\theta_t \leftarrow \Pi_\Theta \left(\theta_{t-1} - \eta \mathbb{E}_{i\sim P_{t-1}} \left[\nabla_\theta \ell_{\text{CE}}(Y_i \langle X_i, \theta_{t-1}\rangle)\right]\right), \; P_t(i) = \frac{1}{-\gamma \sum_{s=1}^{t} \ell_{\text{CE}}(Y_i \langle X_i, \theta_s\rangle) + Z_t}.$$

where Z_t is the normalization constant which ensures $\sum_{i=1}^{n} P_t(i) = 1$. Algorithm 2 in the Appendix describes this procedure. This suggests the following re-weighting of points in Equation equation 3: $P_t(X_i) = \frac{1}{-\gamma \sum_{s=1}^{t} \ell(X_i, Y_i; \theta_s) + Z_t}$. However, this re-weighting is computationally expensive as it requires computing the losses of all the points in the minibatch w.r.t all the iterates $\{\theta_s\}_{s=1}^{t}$. To make this update more tractable, we replace $\sum_{s=1}^{t} \ell(X_i, Y_i; \theta_s)$ in the denominator with $\ell(X_i, Y_i; \theta_t)$. This gives us the following re-weighting scheme: $P_t(X_i) = \frac{1}{-\gamma \ell(X_i, Y_i; \theta_t) + Z_t}$ This suggests picking the following weighting function $g(x) = \frac{1}{1 - \frac{\min(x, M)}{M+1}}$. Here M acts as a normalization constant.

Negative Entropy Regularizer: Picking $\mathcal{R}(P) = \sum_{i=1}^{n} P(i) \log P(i)$, we get: $P_t(i) = \exp(\gamma \sum_{s=1}^{t} \ell_{\text{CE}}(Y_i \langle X_i, \theta_s\rangle) - Z_t)$, for some Z_t which ensures normalization. This suggests the weight function $g(x) = \exp(\max(x, M))$ for some $M > 0$.

3 Experiments

Our proposed solution can be applied to various domains and shows improvements over SOTA approaches in Class Imbalance (**+0.79%**), Corrupted Label (**+2.33%**), Tabular Representation Learning (**+1.44%**), and DomainBed (**+0.7%**).

3.1 Class Imbalance Experiments

This section uses the Long-Tailed CIFAR dataset Cui et al. (2019), reducing the number of training samples per class according to an exponential function, and a ResNet-32 architecture for training. Besides Cross Entropy loss, we also evaluate Focal Loss (Lin et al., 2017) and Class Balanced Loss (Cui et al., 2019) as baselines. Our proposed approach outperforms both Focal Loss and Class Balanced Loss and can be combined with any other loss. Our approach also shows significant improvements on the long-tailed CIFAR-100 dataset. Results and accuracy scores are provided in Appendix 11.1. In comparison to the SOTA (Class Balanced Loss), our approach improves by **+0.79%**. A comparison with L2RW (Ren et al., 2018) and Meta-Weight-Net (Shu et al., 2019) is illustrated in Table 5, where our approach outperforms L2RW and is roughly competitive with Meta-Weight-Net.

3.2 Corrupted Label Experiments

We study the use of *inv*RGD weighting for domains with corrupted labels. The idea is that low-error samples, less likely to be corrupted, should be learned from more. We conduct experiments on CIFAR-10 and CIFAR-100 with flip noise, using ResNet-32, and present basic results in Table 1. Compared to state-of-the-art methods, including Reed-Hard Reed et al. (2014), S-Model Goldberger & Ben-Reuven (2016), Self-Paced Learning Kumar et al. (2010), MentorNet Jiang et al. (2018), and Meta-Weight-Net Shu et al. (2019), we show a **+2.33%** improvement for those without additional data and a **+0.12%** improvement for those with additional data in Appendix 11.2. No additional data is used in our approach.

Table 2: Top table presents results on benchmark binary tabular datasets (AUROC). Bottom table presents results for multi-class datasets(Accuracy). The bottom partition of each table shows performance of RGD. Our reweighting significantly outperforms existing methods, as well as SOTA.

Algorithm	FMNIST	CIFAR10	MNIST	CovType	**Avg.**
MLP	87.62	16.50	96.95	65.47	66.64
RF Breiman (2001)	88.43	42.73	97.62	71.37	75.04
MET Majmundar et al. (2022)	**91.68**	47.82	99.19	76.71	78.85
MET-S					
Default Majmundar et al. (2022)	90.94	48.00	99.01	74.11	78.02
RGD-1 **(Ours)**	91.12	49.17	99.28	79.41	79.75
RGD-EXP **(Ours)**	91.54	**49.54**	**99.69**	**79.72**	**80.12**

Algorithm	Obesity	Income	Criteo	Thyroid	**Avg.**
MLP	52.3	89.39	79.82	62.3	70.95
RF Breiman (2001)	64.36	91.53	77.57	99.62	83.27
MET-S					
Default Majmundar et al. (2022)	71.84	93.85	86.17	99.81	87.92
RGD-1 **(Ours)**	76.23	93.90	86.92	99.82	89.22
RGD-EXP **(Ours)**	**76.87**	**93.96**	**86.98**	**99.92**	**89.43**

3.3 TABULAR REPRESENTATION LEARNING

Our approach improves accuracy in multi-class classification and AUROC performance in binary classification tasks on tabular datasets, compared to previous state-of-the-art. We integrate RGD with MET-S (representation learning without adversarial training) from Majmundar et al. (2022), instead of adversarial training. Our results show **+1.37%** improvement in multi-class classification and **+1.5%** improvement in binary classification, compared to previous SOTA. We compare with more baselines in the appendix. Our experiments on "permuted" MNIST, "permuted" CIFAR, and "permuted" FMNIST were motivated by recent works in tabular representation learning, including Majmundar et al. (2022).

3.4 OUT OF DOMAIN GENERALIZATION

We tested the robustness of a max-margin classifier to distribution shifts (e.g. real vs cartoon pictures) using DomainBed benchmark (Appendix 10). For a long time, the simplest approach, Empirical Risk Minimization (ERM) was the SOTA method (Gulrajani & Lopez-Paz (2020)). However, recent breakthroughs such as MIRO (Cha et al. (2022)) and FRR (Addepalli et al. (2022)) have improved the benchmarks significantly. Our proposed approach RGD integrated with FRR shows improved performance (**+0.7%** avg) as shown in Table3. A more comprehensive comparison with other baselines (IRM Arjovsky et al. (2019), CORAL Sun & Saenko (2016), MTL Blanchard et al. (2021), SagNet Nam et al. (2021)) can be found in Table 10. The environment-wise accuracy of each baseline is in Appendix 11.4.

Table 3: Results on DomainBed (Model selection: training-domain validation set): The bottom partition shows results of our method with RGD loss. In both cases, with (top) and without (bottom) fixed linear layer, the proposed approach outperforms existing methods, as well as SOTA.

Algorithm	PACS	VLCS	OfficeHome	DomainNet	**Avg.**
ERM Gulrajani & Lopez-Paz (2020)	85.5 $\pm$ 0.1	77.5 $\pm$ 0.4	66.5 $\pm$ 0.2	40.9 $\pm$ 0.1	67.6
MIRO Cha et al. (2022)	85.4 $\pm$ 0.4	**79.0** $\pm$ 0.0	70.5 $\pm$ 0.4	44.3 $\pm$ 0.2	69.8
ERM + FRR-L					
Default Addepalli et al. (2022)	85.7 $\pm$ 0.1	76.6 $\pm$ 0.2	68.4 $\pm$ 0.2	44.2 $\pm$ 0.1	68.73
RGD-1 **(Ours)**	87.6 $\pm$ 0.3	78.6 $\pm$ 0.3	69.8 $\pm$ 0.2	**46.00** $\pm$ 0.0	70.48
RGD-EXP **(Ours)**	87.2 $\pm$ 0.3	78.6 $\pm$ 0.3	69.4 $\pm$ 0.2	45.8 $\pm$ 0.0	70.25
ERM + FRR					
Default Addepalli et al. (2022)	87.5 $\pm$ 0.1	77.6 $\pm$ 0.3	69.4 $\pm$ 0.1	45.1 $\pm$ 0.1	69.90
RGD-1 **(Ours)**	**88.2** $\pm$ 0.2	78.6 $\pm$ 0.3	69.8 $\pm$ 0.2	45.8 $\pm$ 0.0	**70.60**
RGD-EXP **(Ours)**	87.6 $\pm$ 0.3	78.1 $\pm$ 0.1	**69.9** $\pm$ 0.1	45.8 $\pm$ 0.0	70.35

4 CONCLUSION

Classical machine learning techniques are very insightful and usually mathematically rigorous. By revisiting these algorithms, and their basics, we can improve our current state-of-the-art deep learning methods in various domains as highlighted in Section 3. We adapted mirror descent-based algorithms for max-margin binary classification in this work to obtain MAX-MARGIN INSPIRED RE-WEIGHTED GRADIENT DESCENT (RGD). This improves performance significantly in tasks requiring robustness, as expected from a max-margin classifier. Our approach is advantageous since the modifications are straightforward and can be easily implemented with a single line of code on top of existing algorithms. While we obtained robustness to noisy labels separately in an ad-hoc way, in future work, we plan to deal with noise tolerance and class imbalance via a single framework. This also requires novel algorithm design and theoretical analysis. We also want to evaluate this technique in various tasks and understand its utility and limitations.

REFERENCES

Sravanti Addepalli, Anshul Nasery, R Venkatesh Babu, Praneeth Netrapalli, and Prateek Jain. Learning an invertible output mapping can mitigate simplicity bias in neural networks. *arXiv preprint arXiv:2210.01360*, 2022.

Sercan Ömer Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. arxiv. *arXiv preprint arXiv:2004.13912*, 2019.

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

Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of computing*, 8(1):121–164, 2012.

Peter Bartlett. For valid generalization the size of the weights is more important than the size of the network. *Advances in neural information processing systems*, 9, 1996.

Peter Bartlett, Yoav Freund, Wee Sun Lee, and Robert E Schapire. Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, 26(5):1651–1686, 1998.

Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Gilles Blanchard, Aniket Anand Deshmukh, Ürun Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *The Journal of Machine Learning Research*, 22(1): 46–100, 2021.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Andy Brock, Soham De, Samuel L Smith, and Karen Simonyan. High-performance large-scale image recognition without normalization. In *International Conference on Machine Learning*, pp. 1059–1071. PMLR, 2021.

Thibault Castells, Philippe Weinzaepfel, and Jerome Revaud. Superloss: A generic loss for robust curriculum learning. *Advances in Neural Information Processing Systems*, 33:4308–4319, 2020.

Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. Domain generalization by mutual-information regularization with pre-trained models. *arXiv preprint arXiv:2203.10789*, 2022.

Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9268–9277, 2019.

Fernando De La Torre and Michael J Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1):117–142, 2003.

Qi Dong, Shaogang Gong, and Xiatian Zhu. Class rectification hard mining for imbalanced deep learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1851–1860, 2017.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Dylan J Foster, Zhiyuan Li, Thodoris Lykouris, Karthik Sridharan, and Eva Tardos. Learning in games: Robustness of fast convergence. *Advances in Neural Information Processing Systems*, 29, 2016.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

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

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. 2016.

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

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

Elad Hazan. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. *Advances in neural information processing systems*, 31, 2018.

Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *European Conference on Computer Vision*, pp. 124–140. Springer, 2020.

Lu Jiang, Deyu Meng, Teruko Mitamura, and Alexander G Hauptmann. Easy samples first: Self-paced reranking for zero-example multimedia search. In *Proceedings of the 22nd ACM international conference on Multimedia*, pp. 547–556, 2014a.

Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander Hauptmann. Self-paced learning with diversity. *Advances in neural information processing systems*, 27, 2014b.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pp. 2304–2313. PMLR, 2018.

Herman Kahn and Andy W Marshall. Methods of reducing sample size in monte carlo computations. *Journal of the Operations Research Society of America*, 1(5):263–278, 1953.

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

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in neural information processing systems*, 23, 2010.

Quoc V Le, Jiquan Ngiam, Adam Coates, Abhik Lahiri, Bobby Prochnow, and Andrew Y Ng. On optimization methods for deep learning. In *Proceedings of the 28th international conference on international conference on machine learning*, pp. 265–272, 2011.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018a.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5400–5409, 2018b.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639, 2018c.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah Erfani, Shutao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. In *International Conference on Machine Learning*, pp. 3355–3364. PMLR, 2018.

Kushal Majmundar, Sachin Goyal, Praneeth Netrapalli, and Prateek Jain. Met: Masked encoding for tabular data. *arXiv preprint arXiv:2206.08564*, 2022.

Tomasz Malisiewicz, Abhinav Gupta, and Alexei A Efros. Ensemble of exemplar-svms for object detection and beyond. In *2011 International conference on computer vision*, pp. 89–96. IEEE, 2011.

Llew Mason, Peter L Bartlett, and Jonathan Baxter. Improved generalization through explicit optimization of margins. *Machine Learning*, 38(3):243–255, 2000.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8690–8699, 2021.

Ali Rahimi and Benjamin Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. *Advances in neural information processing systems*, 21, 2008.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International conference on machine learning*, pp. 4334–4343. PMLR, 2018.

Martin Riedmiller and Heinrich Braun. A direct adaptive method for faster backpropagation learning: The rprop algorithm. In *IEEE international conference on neural networks*, pp. 586–591. IEEE, 1993.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

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

J Schulman, B Zoph, C Kim, J Hilton, J Menick, J Weng, JFC Uribe, L Fedus, L Metz, M Pokorny, et al. Chatgpt: Optimizing language models for dialogue, 2022.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pp. 443–450. Springer, 2016.

Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. Subtab: Subsetting features of tabular data for self-supervised representation learning. *Advances in Neural Information Processing Systems*, 34:18853–18865, 2021.

Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, et al. Applications of machine learning in drug discovery and development. *Nature reviews Drug discovery*, 18(6):463–477, 2019.

Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc Le. Towards domain-agnostic contrastive learning. In *International Conference on Machine Learning*, pp. 10530–10541. PMLR, 2021.

Yixin Wang, Alp Kucukelbir, and David M Blei. Robust probabilistic modeling with bayesian data reweighting. In *International Conference on Machine Learning*, pp. 3646–3655. PMLR, 2017.

Lijun Wu, Fei Tian, Yingce Xia, Yang Fan, Tao Qin, Lai Jian-Huang, and Tie-Yan Liu. Learning to teach with dynamic loss functions. *Advances in neural information processing systems*, 31, 2018.

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

Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. Vime: Extending the success of self-and semi-supervised learning to tabular domain. *Advances in Neural Information Processing Systems*, 33:11033–11043, 2020.

Bianca Zadrozny. Learning and evaluating classifiers under sample selection bias. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 114, 2004.

Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Dinghuai Zhang, Hongyang Zhang, Aaron Courville, Yoshua Bengio, Pradeep Ravikumar, and Arun Sai Suggala. Building robust ensembles via margin boosting. *arXiv preprint arXiv:2206.03362*, 2022.

Marvin Zhang, Henrik Marklund, Nikita Dhawan, Abhishek Gupta, Sergey Levine, and Chelsea Finn. Adaptive risk minimization: Learning to adapt to domain shift. *Advances in Neural Information Processing Systems*, 34:23664–23678, 2021.

Zizhao Zhang and Tomas Pfister. Learning fast sample re-weighting without reward data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 725–734, 2021.

Lanyun Zhu, Tianrun Chen, Jianxiong Yin, Simon See, and Jun Liu. Reinforced sample reweighting policy for semi-supervised learning.

APPENDIX

5 REPRODUCIBILITY STATEMENT

Our proposed loss function is a single line of change. However, one would have to play around with a given task's learning rate (generally lower than the initial setting). Our experiments are based on public datasets and open-source code repositories. The proposed final formulation RGD-1 and RGD-EXP requires **one line of code change**.

Suppose the per-sample loss is given. Example code for applying RGD-1 is shown below.

```
def rgd_t(loss, temp=alpha, reduce=True):
  # alpha >0.
  out = loss * (
      (1 - torch.clamp(loss.detach(), min=0, max=temp) / (temp + 1)) ** (-1)
  )
  return out.sum() / len(out) if reduce else out
```

Similarly, for applying RGD-EXP per-sample weighting, we could use the following modification.

```
def rgd_e(loss, temp=alpha, reduce=True):
    # alpha >0.
  out = loss * torch.exp(
      torch.clamp(loss.detach(), min=0, max=temp) / (temp + 1)
  )
  return out.sum() / len(out) if reduce else out
```

6 RELATED WORKS

In this section, we describe prior studies that are broadly related to our problem at hand. Below, we divide prior research under three prominent subheadings: (a) Per-Sample Re-weighting, (b) Pre-Conditioning, and (c) Max-Margin Learning.

6.1 PER-SAMPLE REWEIGHTING

The idea of re-weighting samples can be dated back to the works of Chawla et al. (2002); Dong et al. (2017); Zadrozny (2004). These works involved pre-evaluating per-sample weights as a pre-processing step using some prior knowledge. To alleviate this need for human domain supervision, recent approaches have moved towards dynamically computing the per-sample weights using another class of functions, such as a neural network. Many approaches, such as Freund & Schapire (1997); Kumar et al. (2010); Kahn & Marshall (1953), have considered using necessary samples for better model training. Other works, such as Wu et al. (2018), proposed a dynamic loss function that works on the intuition of using a student network to learn the task at hand and a teacher network that updates the logits from the student network. Both networks are trained in an interleaved fashion. Here, the per-sample re-weighting takes place at the logit level instead of the typical loss level. Recently, there has been a growing research interest in using meta-learning and reinforcement learning based-approaches for re-weighting samples. For instance, Ren et al. (2018); Shu et al. (2019) use meta-learning methods such as MAML to output weights of each sample. Another work uses a history buffer which stores a snapshot of the trajectory of each point, facilitating giving more importance to points which led to more learning in our model Zhang & Pfister (2021). Other approaches, such as Zhu et al., use reinforcement learning to learn the per-sample weights using a "pretraining-boosting" two-stage MDP curriculum where the agent network is firstly pre-trained and optimized for deployment in the classification problem.

There are predominantly two paradigms for re-weighting samples:

- Works such as Freund et al. (1999); Malisiewicz et al. (2011); Lin et al. (2017) consider re-weighting samples as monotonically increasing functions of the loss helps emphasize

samples with larger loss values. These samples are more likely to be uncertain, complex samples close to our classification model's decision boundary.

- Other approaches such as Kumar et al. (2010); De La Torre & Black (2003); Jiang et al. (2014a;b); Wang et al. (2017) re-weight the samples as a monotonically decreasing function of the loss to take samples with smaller loss values as important ones. This rationality lies in the idea that these low loss-value samples are more likely to be confident with clean labels, thus allowing the model to learn better.

Furthermore, works such Castells et al. (2020) propose a confidence-aware loss proportional to the lowest loss of that sample. They also use a threshold (γ) to decide how practical each point is, i.e., the importance the easy and hard samples should have instead of being equal. In this work, we consider both of these scenarios.

6.2 PRE-CONDITIONING

Generally, pre-conditioning can be the normalization of inputs, batch normalization, scaling gradients in a few directions, and other approaches that help aid learning. This work predominantly discusses the sub-domain, which focuses on scaling gradients in a few directions for efficient backpropagation. Although deep learning presents various computational challenges, such as the non-convex learning objectives, prior works have studied their behavior and come up with various solutions to alleviate some of these limitations. A common technique to improve the efficiency of our model is to use adaptive step-size methods for optimization, such as the Newton method, which takes advantage of the second-order gradients. However, computing the Hessian matrix is computationally intensive, leading to Quasi-Newton methods: methods that approximate the value of the Hessian instead of computing them every time Le et al. (2011). Another popular alternative is to use an element-wise adaptive learning rate, which has shown great promise in ADAgrad Duchi et al. (2011), RMSProp Ruder (2016), ADAdelta Zeiler (2012). For instance, ADAgrad is a diagonal pre-conditioning technique where the pre-conditioning across each dimension is computed as the inverse square root of the norms of gradients along that dimension accumulated over training. Unfortunately, due to this accumulation of gradients, it is often susceptible to falling in a saddle point as the scaling factor decreases monotonically. Another important work is RProp by Riedmiller & Braun (1993), which, unlike ADAgrad, does not worry about the magnitude of gradients. Instead of taking different step sizes across various dimensions, it only uses the gradient sign, thus guaranteeing that each dimension will have weight updates of the same step size.

6.3 MAX-MARGIN LEARNING AND MIRROR DESCENT

Margin is a fundamental machine learning notion that is known to govern the generalization performance of a model (Bartlett, 1996). Max-margin learning has been attributed to the success of popular ML techniques such as SVMs and boosting (Bartlett et al., 1998; Mason et al., 2000). Max margin classifiers (i.e., where the decision boundary is chosen to be far away from all the points being classified) are favored for their robustness, and generalization since small perturbations in the data does not lead to misclassification. These ideas are also helpful in designing and understanding the performance of deep learning algorithms. For instance, max-margin learning has been used to design learning algorithms robust to adversarial corruptions (Zhang et al., 2022). In this work, we derive a re-weighting scheme based on this idea.

Max-margin learning has a game-theoretic flavor to it. In particular, the objective function can be written as a min-max problem, where we want to learn a classifier with the best worst-case margin. *Mirror descent* is a popular algorithm often used to solve such games. The optimization community originally proposed Mirror descent as a generalization to gradient descent (Beck & Teboulle, 2003). Recent works have shown its utility for solving min-max games (Hazan, 2016; Cesa-Bianchi & Lugosi, 2006). The re-weighting schemes we derive in our work are developed using mirror descent with negative entropy and log-barrier regularizers.

Our approach has close connections to boosting. In boosting, we learn an ensemble of classifiers that can perform well on the learning task. Popular boosting algorithms such as AdaBoost (Arora et al., 2012) rely on the exponential re-weighting mechanism, which is built using mirror descent with

entropy regularizer, to learn the ensembles. However, unlike AdaBoost, we do not build ensembles. Our algorithm learns a single model that does well on the learning task.

7 Proof of Lemma 1

Proof. Notice that for every X and θ, we have: $\int Q(dY|X)\ell(X,Y;\theta) \geq \int Q(dY|X)\ell(X,Y;\theta^*)$. Hence, by monotonicity of expectation, $\int P(dX;\theta)\left[\int Q(dY|X)\ell(X,Y;\theta)\right] \geq \int P(dX;\theta)\left[\int Q(dY|X)\ell(X,Y;\theta^*)\right]$ □

8 Proof of Lemma 2

Proof. This follows from the fact that $\nabla_\theta \ell(X,Y;\theta^*) = 0$ for every X. □

9 Algorithms for Max-Margin Learning

Algorithm 2 Max-Margin Learning using Online Mirror Descent with log-barrier regularizer

1: **Input:** Data $\{X_i, Y_i\}_{i=1}^n$, learning rates η, γ for min and max players
2: **for** $t = 1 \dots T$ **do**
3: Update θ_t using projected gradient descent

$$\theta_t \leftarrow \Pi_\Theta\left(\theta_{t-1} - \eta \mathbb{E}_{i\sim P_{t-1}}\left[\nabla_\theta \ell_{\text{CE}}(Y_i \langle X_i, \theta_{t-1}\rangle)\right]\right)$$

4: Update P_t using online mirror descent with log-barrier regularizer

$$P_t = \arg\max_{P\in\Delta_n} \sum_{s=1}^{t} \mathbb{E}_{i\sim P}\left[\ell_{\text{CE}}(Y_i \langle X_i, \theta_s\rangle)\right] + \frac{1}{\gamma}\sum_{i=1}^{n} \log P(i)$$

The solution of this optimization problem has the following analytical expression

$$P_t(i) = \frac{1}{-\gamma \sum_{s=1}^{t} \ell_{\text{CE}}(Y_i \langle X_i, \theta_s\rangle) + Z_t},$$

where Z_t is the normalization constant which ensures $\sum_{i=1}^n P_t(i) = 1$.
5: **end for**
6: **Output:** $\frac{1}{T}\sum_{t=1}^{T} \theta_t$.

10 DomainBed Benchmark

In this section, we describe the DomainBed benchmark, a challenging benchmark used to study the out-of-domain generalization capabilities of our model. To briefly explain, consider the dataset `PACS`, which consists of Photos, Art, cartoons, and sketches of the same set of classes (for instance, dogs, and cats, amongst others). The goal of the task is to learn from three of these domains and evaluate the performance of the left-out domain (similar to a k-fold cross-validation). By doing so, we can evaluate the out-of-domain generalization performance of our models. In general, the metric used in this domain involves taking an average of the performance of the different k-fold splits. More information about this benchmark is available at Gulrajani & Lopez-Paz (2020).

11 Additional Results

This section briefly discusses more elaborated tables and findings from our research.

11.1 Class Imbalance Experiments

This section briefly discusses additional results from our experiments on the Class Imbalance domain with datasets such as CIFAR-10 and CIFAR-100. Table 4 depicts the accuracy metric of models

on various levels of the imbalance factor. From Table 4, we show that our proposed approach RGD-EXP outperforms other baselines such as Focal Loss and Class Balanced Loss by **+0.79%**. Furthermore, when models are trained on additional data, either by fine-tuning or by using a meta-learning framework to learn weights (such as Meta-Weight-Net and L2RW), we show that our proposed approach is competitively similar (**-0.22%**). Table 5 illustrates this analysis further. The performance metrics of the baseline approaches were taken from Shu et al. (2019).

Table 4: Test Accuracy of ResNet-32 on Long-Tailed CIFAR-10, and CIFAR-100 dataset.

Dataset	CIFAR-10							CIFAR-100						
Loss / Imbalance Factor	200	100	50	20	10	1	**Avg.**	200	100	50	20	10	1	**Avg.**
Focal Loss Lin et al. (2017)	65.29	70.38	76.71	82.76	86.66	93.03	79.14	35.62	38.41	44.32	51.95	55.78	70.52	49.43
Class Balanced Loss Cui et al. (2019)	**68.89**	**74.57**	79.27	84.36	87.49	92.89	81.25	36.23	39.60	45.32	52.59	57.99	70.50	50.21
Cross Entropy (CE)														
Default	65.98	70.36	74.81	82.23	86.39	92.89	78.78	34.84	38.32	43.85	51.14	55.71	70.50	49.06
RGD-1 **(Ours)**	64.16	72.56	77.86	83.88	86.84	92.99	79.72	36.22	39.87	43.74	51.86	56.9	70.80	49.90
RGD-EXP **(Ours)**	67.90	73.75	**79.63**	**85.44**	**88.00**	**93.27**	**81.33**	**38.62**	**41.89**	**46.40**	**53.48**	**58.5**	**71.30**	**51.70**

Table 5: Test Accuracy of ResNet-32 on Long-Tailed CIFAR-10, and CIFAR-100 dataset. We use the symbol $\star$ to denote approaches that use additional data (as the meta-dataset). We use $\underline{underline}$ symbol to depict performances which are second-best across baselines. Our experiments show that we can get competitively similar performance to such models as well without training a second neural network.

Dataset	CIFAR-10							CIFAR-100						
Loss / Imbalance Factor	200	100	50	20	10	1	**Avg.**	200	100	50	20	10	1	**Avg.**
Fine-tuning $\star$	66.08	71.33	77.42	83.37	86.42	93.23	79.64	38.22	41.83	46.40	52.11	57.44	70.72	51.12
L2RW Ren et al. (2018) $\star$	66.51	74.16	78.93	82.12	85.19	89.25	77.69	33.38	40.23	44.44	51.64	53.73	64.11	47.92
Meta-Weight-Net Shu et al. (2019) $\star$	**68.91**	**75.21**	**80.06**	84.94	87.84	92.66	**81.60**	37.91	**42.09**	**46.74**	**54.37**	58.46	70.37	51.65
Cross Entropy (CE)														
Default	65.98	70.36	74.81	82.23	86.39	92.89	78.78	34.84	38.32	43.85	51.14	55.71	70.50	49.06
RGD-1 **(Ours)**	64.16	72.56	77.86	83.88	86.84	92.99	79.72	36.22	39.87	43.74	51.86	56.9	70.80	49.90
RGD-EXP **(Ours)**	67.90	73.75	79.63	**85.44**	**88.00**	**93.27**	81.33	**38.62**	41.89	46.40	53.48	**58.5**	**71.30**	**51.70**

11.2 CORRUPTED LABEL EXPERIMENTS

This section briefly discusses additional results from our experiments on label corruption on datasets such as CIFAR-10 and CIFAR-100. Table 6 depicts our proposed approach's accuracy metric compared to various state-of-the-art baselines. From the above table, we show that our proposed *inv* RGD-EXP and *inv* RGD-1 approach outperforms other state-of-the-art baselines which do not use additional data by **+2.33%**. Furthermore, our approach is competitively similar (**+0.12%**) to approaches that use additional data (using meta-learning) as shown in Table 7. The performance metrics of the baseline approaches were taken from Shu et al. (2019).

Table 6: Results on CIFAR-10 and CIFAR-100 dataset with flip noise.

Dataset	CIFAR-10				CIFAR-100			
Loss	0%	20%	40%	**Avg.**	0%	20%	40%	**Avg.**
Reed-Hard Reed et al. (2014)	92.31 $\pm$ 0.25	88.28 $\pm$ 0.36	81.06 $\pm$ 0.76	87.22	69.02 $\pm$ 0.32	60.27 $\pm$ 0.76	50.40 $\pm$ 1.01	59.90
S-Model Goldberger & Ben-Reuven (2016)	83.61 $\pm$ 0.13	79.25 $\pm$ 0.30	75.73 $\pm$ 0.32	79.53	51.46 $\pm$ 0.20	45.45 $\pm$ 0.25	43.81 $\pm$ 0.15	46.91
Self-Paced Kumar et al. (2010)	88.52 $\pm$ 0.21	87.03 $\pm$ 0.34	81.63 $\pm$ 0.52	85.73	67.55 $\pm$ 0.27	63.63 $\pm$ 0.30	53.51 $\pm$ 0.53	61.56
Focal Loss Lin et al. (2017)	93.03 $\pm$ 0.16	86.45 $\pm$ 0.19	80.45 $\pm$ 0.97	86.64	70.02 $\pm$ 0.53	61.87 $\pm$ 0.30	54.13 $\pm$ 0.40	62.01
Co-Teaching Han et al. (2018)	89.87 $\pm$ 0.10	82.83 $\pm$ 0.85	75.41 $\pm$ 0.21	82.70	63.31 $\pm$ 0.05	54.13 $\pm$ 0.55	44.85 $\pm$ 0.81	54.20
D2L Ma et al. (2018)	92.02 $\pm$ 0.14	87.66 $\pm$ 0.40	83.89 $\pm$ 0.46	87.86	68.11 $\pm$ 0.26	63.48 $\pm$ 0.53	51.83 $\pm$ 0.33	61.14
Cross Entropy (CE)								
Default	92.89 $\pm$ 0.32	76.83 $\pm$ 2.30	70.77 $\pm$ 2.31	80.16	70.50 $\pm$ 0.12	50.86 $\pm$ 0.27	43.01 $\pm$ 1.16	54.79
inv RGD-1 **(Ours)**	93.13 $\pm$ 0.17	90.91 $\pm$ 0.21	86.05 $\pm$ 0.28	90.03	71.31 $\pm$ 0.21	67.09 $\pm$ 0.23	54.06 $\pm$ 0.38	64.15
inv RGD-EXP **(Ours)**	**93.21** $\pm$ 0.26	**91.19** $\pm$ 0.14	**86.39** $\pm$ 0.61	**90.26**	**71.40** $\pm$ 0.18	**67.32** $\pm$ 0.24	**54.08** $\pm$ 0.77	**64.27**

Table 7: Results on CIFAR-10 and CIFAR-100 dataset with flip noise. We use the symbol ⋆ to denote approaches that use additional data (as the meta-dataset). We use *underline* symbol to depict performances which are second-best across baselines. Our experiments show that we can get competitively similar performance to such models as well without training a second neural network.

Dataset	CIFAR-10				CIFAR-100			
Loss	0%	20%	40%	**Avg.**	0%	20%	40%	**Avg.**
Fine-Tuning ⋆	93.23 ± 0.23	82.47 ± 3.64	74.07 ± 1.56	83.26	70.72 ± 0.22	56.98 ± 0.50	46.37 ± 0.25	58.02
MentorNet Jiang et al. (2018) ⋆	92.13 ± 0.30	86.36 ± 0.31	81.76 ± 0.28	86.75	70.24 ± 0.21	61.97 ± 0.47	52.66 ± 0.56	61.62
L2RW Ren et al. (2018)	89.25 ± 0.37	87.86 ± 0.36	85.66 ± 0.51	87.59	64.11 ± 1.09	57.47 ± 1.16	50.98 ± 1.55	57.52
GLC Hendrycks et al. (2018) ⋆	91.02 ± 0.20	89.68 ± 0.33	**88.92** ± 0.24	89.87	65.42 ± 0.23	63.07 ± 0.53	**62.22** ± 0.62	63.57
Meta-Weight-Net Shu et al. (2019) ⋆	92.04 ± 0.15	90.33 ± 0.61	87.54 ± 0.23	89.97	70.11 ± 0.33	64.22 ± 0.28	58.64 ± 0.47	**64.32**
Cross Entropy (CE)								
Default	92.89 ± 0.32	76.83 ± 2.30	70.77 ± 2.31	80.16	70.50 ± 0.12	50.86 ± 0.27	43.01 ± 1.16	54.79
inv RGD-1 **(Ours)**	93.13 ± 0.17	90.91 ± 0.21	86.05 ± 0.28	90.03	71.31 ± 0.21	67.09 ± 0.23	54.06 ± 0.38	64.15
inv RGD-EXP **(Ours)**	**93.21** ± 0.26	**91.19** ± 0.14	86.39 ± 0.61	**90.26**	**71.40** ± 0.18	**67.32** ± 0.24	54.08 ± 0.77	64.27

Table 8: Results on standard multi-class tabular datasets (Accuracy): The bottom partition shows results of our method with RGD loss. We show that the addition of our proposed approach significantly outperforms existing methods, as well as SOTA.

Algorithm	FMNIST	CIFAR10	MNIST	CovType	**Avg.**
MLP	87.62	16.50	96.95	65.47	66.64
RF Breiman (2001)	88.43	42.73	97.62	71.37	75.04
GBDT Friedman (2001)	88.71	45.7	**100**	72.96	76.84
RF-G Rahimi & Recht (2008)	89.84	29.32	97.65	71.57	72.10
MET-R Majmundar et al. (2022)	88.84	28.94	97.44	69.68	71.23
VIME Yoon et al. (2020)	80.36	34.00	95.77	62.80	68.23
DACL+ Verma et al. (2021)	81.40	39.70	91.40	64.23	69.18
SubTab Ucar et al. (2021)	87.59	39.34	98.31	42.36	66.90
TabNet Arik & Pfister (2019)	88.18	33.75	96.63	65.13	70.92
MET Majmundar et al. (2022)	91.68	47.82	99.19	76.71	78.85
MET-S					
Default Majmundar et al. (2022)	90.94	48.00	99.01	74.11	78.02
RGD-1 **(Ours)**	91.12	49.17	99.28	79.41	79.75
RGD-EXP **(Ours)**	**91.54**	**49.54**	99.69	**79.72**	**80.12**

11.3 TABULAR REPRESENTATION LEARNING

This section discusses a few additional results from our experiments on Tabular Representation Learning. Table 8 depicts our proposed approach's accuracy compared to other baselines on multi-class tabular datasets. Our approach outperforms previous SOTA in this problem by **+1.37%**. Furthermore, Table 9 illustrates the AUROC score of our proposed approach in comparison to state-of-the-art baselines on binary-class tabular datasets. Our approach shows an improvement of **+1.5%** in this setting as well. The performance metrics of the baseline approaches were taken from Majmundar et al. (2022).

11.4 DOMAINBED

In this section, we briefly discuss additional results from our DomainBed experiments. Table 10 depicts a complete table and comparison of our proposed approach to a multitude of state-of-the-art approaches in this field. Furthermore, we also show that our proposed approach outperforms previous SOTA by **+0.7%**. Furthermore, we also present the per-environment breakdown of our approach in various datasets in Table 11, Table 12, Table 13, and Table 14 for PACS, VLCS, OfficeHome, and DomainNet respectively. The performance metrics of the baseline approaches were taken from Gulrajani & Lopez-Paz (2020).

Table 9: Results on standard binary-class tabular datasets (AUROC): The bottom partition shows results of our method with RGD loss. We show that the addition of our proposed approach significantly outperforms existing methods, as well as SOTA.

Algorithm	Obesity	Income	Criteo	Thyroid	**Avg.**
MLP	52.3	89.39	79.82	62.3	70.95
RF Breiman (2001)	64.36	91.53	77.57	99.62	83.27
GBDT Friedman (2001)	64.4	92.5	78.77	99.34	83.75
RF-G Rahimi & Recht (2008)	54.45	90.09	80.32	52.65	69.37
MET-R Majmundar et al. (2022)	53.2	83.54	79.17	82.03	74.49
VIME Yoon et al. (2020)	57.27	87.37	74.28	94.87	78.45
DACL+ Verma et al. (2021)	61.18	89.01	75.32	86.63	78.04
SubTab Ucar et al. (2021)	64.92	88.95	76.57	88.93	79.00
TabNet Arik & Pfister (2019)	69.40	77.30	80.91	96.98	81.15
MET-S					
Default Majmundar et al. (2022)	71.84	93.85	86.17	99.81	87.92
RGD-1 **(Ours)**	76.23	93.90	86.92	99.82	89.22
RGD-EXP **(Ours)**	**76.87**	**93.96**	**86.98**	**99.92**	**89.43**

Table 10: Results on DomainBed (Model selection: training-domain validation set): The bottom partition shows results of our method with RGD loss. In both cases, with (top) and without (bottom) fixed linear layer, the proposed approach outperforms existing methods, as well as SOTA.

Algorithm	PACS	VLCS	OfficeHome	DomainNet	**Avg.**
ERM Gulrajani & Lopez-Paz (2020)	85.5 $\pm$ 0.1	77.5 $\pm$ 0.4	66.5 $\pm$ 0.2	40.9 $\pm$ 0.1	67.6
IRM Arjovsky et al. (2019)	83.5 $\pm$ 0.8	78.5 $\pm$ 0.5	64.3 $\pm$ 2.2	33.9 $\pm$ 2.8	65.1
GroupDRO Sagawa et al. (2019)	84.4 $\pm$ 0.8	76.7 $\pm$ 0.6	66.0 $\pm$ 0.7	33.3 $\pm$ 0.2	65.1
Mixup Yan et al. (2020)	84.6 $\pm$ 0.6	77.4 $\pm$ 0.6	68.1 $\pm$ 0.3	39.2 $\pm$ 0.1	67.33
MLDG Li et al. (2018a)	84.9 $\pm$ 1.0	77.2 $\pm$ 0.4	66.8 $\pm$ 0.6	41.2 $\pm$ 0.1	67.53
CORAL Sun & Saenko (2016)	86.2$\pm$ 0.3	78.8$\pm$ 0.6	68.7$\pm$ 0.3	41.5 $\pm$ 0.1	68.8
MMD Li et al. (2018b)	84.6 $\pm$ 0.5	77.5 $\pm$ 0.9	66.3 $\pm$ 0.1	23.4 $\pm$ 9.5	62.95
DANN Ganin et al. (2016)	83.6 $\pm$ 0.4	78.6 $\pm$ 0.4	65.9 $\pm$ 0.6	38.3 $\pm$ 0.1	66.6
CDANN Li et al. (2018c)	82.6 $\pm$ 0.9	77.5 $\pm$ 0.1	65.8 $\pm$ 1.3	38.3 $\pm$ 0.3	66.05
MTL Blanchard et al. (2021)	84.6 $\pm$ 0.5	77.2 $\pm$ 0.4	66.4 $\pm$ 0.5	40.6 $\pm$ 0.1	67.2
SagNet Nam et al. (2021)	86.3 $\pm$ 0.2	77.8 $\pm$ 0.5	68.1 $\pm$ 0.1	40.3 $\pm$ 0.1	68.13
ARM Zhang et al. (2021)	85.1 $\pm$ 0.4	77.6 $\pm$ 0.3	64.8 $\pm$ 0.3	35.5 $\pm$ 0.2	65.75
VREx Krueger et al. (2021)	84.9 $\pm$ 0.6	78.3 $\pm$ 0.2	66.4 $\pm$ 0.6	33.6 $\pm$ 2.9	65.8
RSC Huang et al. (2020)	85.2 $\pm$ 0.9	77.1 $\pm$ 0.5	65.5 $\pm$ 0.9	38.9 $\pm$ 0.5	66.68
MIRO Cha et al. (2022)	85.4 $\pm$ 0.4	79.0 $\pm$ 0.0	70.5 $\pm$ 0.4	44.3 $\pm$ 0.2	69.8
ERM + FRR-L					
Default Addepalli et al. (2022)	85.7 $\pm$ 0.1	76.6 $\pm$ 0.2	68.4 $\pm$ 0.2	44.2 $\pm$ 0.1	68.73
RGD-1 **(Ours)**	**87.6** $\pm$ 0.3	78.6 $\pm$ 0.3	**69.8** $\pm$ 0.2	**46.0** $\pm$ 0.0	**70.48**
RGD-EXP **(Ours)**	87.2 $\pm$ 0.3	78.6 $\pm$ 0.3	69.4 $\pm$ 0.2	45.8 $\pm$ 0.0	70.25
ERM + FRR					
Default Addepalli et al. (2022)	87.5 $\pm$ 0.1	77.6 $\pm$ 0.3	69.4 $\pm$ 0.1	45.1 $\pm$ 0.1	69.9
RGD-1 **(Ours)**	**88.2** $\pm$ 0.2	78.6 $\pm$ 0.3	69.8 $\pm$ 0.2	45.8 $\pm$ 0.0	**70.6**
RGD-EXP **(Ours)**	87.6 $\pm$ 0.3	78.1 $\pm$ 0.1	**69.9** $\pm$ 0.1	45.8 $\pm$ 0.0	70.35

Table 11: **Out-of-domain accuracies (%) on** PACS**.**

Algorithm	**A**	**C**	**P**	**S**	**Avg**
CDANN	84.6 ± 1.8	75.5 ± 0.9	96.8 ± 0.3	73.5 ± 0.6	82.6
MASF	82.9	80.5	95.0	72.3	82.7
DMG	82.6	78.1	94.5	78.3	83.4
IRM	84.8 ± 1.3	76.4 ± 1.1	96.7 ± 0.6	76.1 ± 1.0	83.5
MetaReg	87.2	79.2	97.6	70.3	83.6
DANN	86.4 ± 0.8	77.4 ± 0.8	97.3 ± 0.4	73.5 ± 2.3	83.7
GroupDRO	83.5 ± 0.9	79.1 ± 0.6	96.7 ± 0.3	78.3 ± 2.0	84.4
MTL	87.5 ± 0.8	77.1 ± 0.5	96.4 ± 0.8	77.3 ± 1.8	84.6
I-Mixup	86.1 ± 0.5	78.9 ± 0.8	97.6 ± 0.1	75.8 ± 1.8	84.6
MMD	86.1 ± 1.4	79.4 ± 0.9	96.6 ± 0.2	76.5 ± 0.5	84.7
VREx	86.0 ± 1.6	79.1 ± 0.6	96.9 ± 0.5	77.7 ± 1.7	84.9
MLDG	85.5 ± 1.4	80.1 ± 1.7	97.4 ± 0.3	76.6 ± 1.1	84.9
ARM	86.8 ± 0.6	76.8 ± 0.5	97.4 ± 0.3	79.3 ± 1.2	85.1
RSC	85.4 ± 0.8	79.7 ± 1.8	97.6 ± 0.3	78.2 ± 1.2	85.2
Mixstyle	86.8 ± 0.5	79.0 ± 1.4	96.6 ± 0.1	78.5 ± 2.3	85.2
ER	87.5	79.3	98.3	76.3	85.3
pAdaIN	85.8	81.1	97.2	77.4	85.4
ERM	84.7 ± 0.4	80.8 ± 0.6	97.2 ± 0.3	79.3 ± 1.0	85.5
EISNet	86.6	81.5	97.1	78.1	85.8
CORAL	88.3 ± 0.2	80.0 ± 0.5	97.5 ± 0.3	78.8 ± 1.3	86.2
SagNet	87.4 ± 1.0	80.7 ± 0.6	97.1 ± 0.1	80.0 ± 0.4	86.3
DSON	87.0	80.6	96.0	82.9	86.6
ERM + FRR-L					
Default	83.2 ± 0.3	79.8 ± 0.4	95.9 ± 0.3	83.5 ± 0.4	85.7
RGD-1 **(Ours)**	88.4 ± 0.3	83.3 ± 0.8	97.5 ± 0.3	81.1 ± 0.5	**87.6**
RGD-EXP **(Ours)**	88.7 ± 0.5	83.0 ± 0.5	97.8 ± 0.1	79.4 ± 1.0	87.2
ERM + FRR					
Default	86.8 ± 0.3	82.2 ± 0.4	96.4 ± 0.1	84.5 ± 0.2	87.5
RGD-1 **(Ours)**	88.8 ± 0.3	84.0 ± 0.8	97.7 ± 0.1	82.4 ± 0.6	**88.2**
RGD-EXP **(Ours)**	87.7 ± 0.8	84.0 ± 0.6	97.6 ± 0.1	81.2 ± 0.5	87.6

Table 12: **Out-of-domain accuracies (%) on** VLCS**.**

Algorithm	**C**	**L**	**S**	**V**	**Avg**
GroupDRO	97.3 ± 0.3	63.4 ± 0.9	69.5 ± 0.8	76.7 ± 0.7	76.7
RSC	97.9 ± 0.1	62.5 ± 0.7	72.3 ± 1.2	75.6 ± 0.8	77.1
MLDG	97.4 ± 0.2	65.2 ± 0.7	71.0 ± 1.4	75.3 ± 1.0	77.2
MTL	97.8 ± 0.4	64.3 ± 0.3	71.5 ± 0.7	75.3 ± 1.7	77.2
I-Mixup	98.3 ± 0.6	64.8 ± 1.0	72.1 ± 0.5	74.3 ± 0.8	77.4
ERM	97.7 ± 0.4	64.3 ± 0.9	73.4 ± 0.5	74.6 ± 1.3	77.5
MMD	97.7 ± 0.1	64.0 ± 1.1	72.8 ± 0.2	75.3 ± 3.3	77.5
CDANN	97.1 ± 0.3	65.1 ± 1.2	70.7 ± 0.8	77.1 ± 1.5	77.5
ARM	98.7 ± 0.2	63.6 ± 0.7	71.3 ± 1.2	76.7 ± 0.6	77.6
SagNet	97.9 ± 0.4	64.5 ± 0.5	71.4 ± 1.3	77.5 ± 0.5	77.8
Mixstyle	98.6 ± 0.3	64.5 ± 1.1	72.6 ± 0.5	75.7 ± 1.7	77.9
VREx	98.4 ± 0.3	64.4 ± 1.4	74.1 ± 0.4	76.2 ± 1.3	78.3
IRM	98.6 ± 0.1	64.9 ± 0.9	73.4 ± 0.6	77.3 ± 0.9	78.6
DANN	99.0 ± 0.3	65.1 ± 1.4	73.1 ± 0.3	77.2 ± 0.6	78.6
CORAL	98.3 ± 0.1	66.1 ± 1.2	73.4 ± 0.3	77.5 ± 1.2	78.8
ERM + FRR-L					
Default	97.1 ± 0.2	63.3 ± 0.3	72.0 ± 0.3	74.3 ± 0.3	76.6
RGD-1 **(Ours)**	98.9 ± 0	64.9 ± 0.4	73.2 ± 0.4	77.5 ± 0.6	78.6
RGD-EXP **(Ours)**	98.8 ± 0.1	64.8 ± 0.2	73.9 ± 0.2	77.0 ± 1.1	78.6
ERM + FRR					
Default	96.7 ± 0.6	65.2 ± 0.8	73.4 ± 0.1	75.6 ± 0.4	77.6
RGD-1 **(Ours)**	97.1 ± 0.5	65.4 ± 0.8	74.3 ± 0.1	77.5 ± 0.3	78.6
RGD-EXP **(Ours)**	98.3 ± 0.1	64.5 ± 0.2	72.3 ± 0.1	77.2 ± 0.3	78.1

Table 13: **Out-of-domain accuracies (%) on** `OfficeHome`**.**

Algorithm	**A**	**C**	**P**	**R**	**Avg**
Mixstyle	51.1 ± 0.3	53.2 ± 0.4	68.2 ± 0.7	69.2 ± 0.6	60.4
IRM	58.9 ± 2.3	52.2 ± 1.6	72.1 ± 2.9	74.0 ± 2.5	64.3
ARM	58.9 ± 0.8	51.0 ± 0.5	74.1 ± 0.1	75.2 ± 0.3	64.8
RSC	60.7 ± 1.4	51.4 ± 0.3	74.8 ± 1.1	75.1 ± 1.3	65.5
CDANN	61.5 ± 1.4	50.4 ± 2.4	74.4 ± 0.9	76.6 ± 0.8	65.7
DANN	59.9 ± 1.3	53.0 ± 0.3	73.6 ± 0.7	76.9 ± 0.5	65.9
GroupDRO	60.4 ± 0.7	52.7 ± 1.0	75.0 ± 0.7	76.0 ± 0.7	66.0
MMD	60.4 ± 0.2	53.3 ± 0.3	74.3 ± 0.1	77.4 ± 0.6	66.4
MTL	61.5 ± 0.7	52.4 ± 0.6	74.9 ± 0.4	76.8 ± 0.4	66.4
VREx	60.7 ± 0.9	53.0 ± 0.9	75.3 ± 0.1	76.6 ± 0.5	66.4
ERM	61.3 ± 0.7	52.4 ± 0.3	75.8 ± 0.1	76.6 ± 0.3	66.5
MLDG	61.5 ± 0.9	53.2 ± 0.6	75.0 ± 1.2	77.5 ± 0.4	66.8
I-Mixup	62.4 ± 0.8	54.8 ± 0.6	76.9 ± 0.3	78.3 ± 0.2	68.1
SagNet	63.4 ± 0.2	54.8 ± 0.4	75.8 ± 0.4	78.3 ± 0.3	68.1
CORAL	65.3 ± 0.4	54.4 ± 0.5	76.5 ± 0.1	78.4 ± 0.5	68.7
ERM + FRR-L					
Default	64.4 ± 0.1	55.6 ± 0.5	76.5 ± 0.2	77.5 ± 0.2	68.4
RGD-1 **(Ours)**	64.5 ± 0.3	56.9 ± 0.5	77.8 ± 0.3	80.0 ± 0.4	**69.8**
RGD-EXP **(Ours)**	64.2 ± 0.3	55.9 ± 0.5	77.6 ± 0.2	79.9 ± 0.3	69.4
ERM + FRR					
Default	64.5 ± 0.2	58.4 ± 0.1	76.6 ± 0.3	78.3 ± 0.1	69.4
RGD-1 **(Ours)**	65.6 ± 0.5	56.9 ± 0.3	76.9 ± 0.1	79.7 ± 0.3	69.8
RGD-EXP **(Ours)**	65.6 ± 0.3	57.1 ± 0.3	76.8 ± 0.3	80.2 ± 0.2	**69.9**

Table 14: **Out-of-domain accuracies (%) on** `DomainNet`**.**

Algorithm	**clip**	**info**	**paint**	**quick**	**real**	**sketch**	**Avg**
MMD	32.1 ± 13.3	11.0 ± 4.6	26.8 ± 11.3	8.7 ± 2.1	32.7 ± 13.8	28.9 ± 11.9	23.4
GroupDRO	47.2 ± 0.5	17.5 ± 0.4	33.8 ± 0.5	9.3 ± 0.3	51.6 ± 0.4	40.1 ± 0.6	33.3
VREx	47.3 ± 3.5	16.0 ± 1.5	35.8 ± 4.6	10.9 ± 0.3	49.6 ± 4.9	42.0 ± 3.0	33.6
IRM	48.5 ± 2.8	15.0 ± 1.5	38.3 ± 4.3	10.9 ± 0.5	48.2 ± 5.2	42.3 ± 3.1	33.9
Mixstyle	51.9 ± 0.4	13.3 ± 0.2	37.0 ± 0.5	12.3 ± 0.1	46.1 ± 0.3	43.4 ± 0.4	34.0
ARM	49.7 ± 0.3	16.3 ± 0.5	40.9 ± 1.1	9.4 ± 0.1	53.4 ± 0.4	43.5 ± 0.4	35.5
CDANN	54.6 ± 0.4	17.3 ± 0.1	43.7 ± 0.9	12.1 ± 0.7	56.2 ± 0.4	45.9 ± 0.5	38.3
DANN	53.1 ± 0.2	18.3 ± 0.1	44.2 ± 0.7	11.8 ± 0.1	55.5 ± 0.4	46.8 ± 0.6	38.3
RSC	55.0 ± 1.2	18.3 ± 0.5	44.4 ± 0.6	12.2 ± 0.2	55.7 ± 0.7	47.8 ± 0.9	38.9
I-Mixup	55.7 ± 0.3	18.5 ± 0.5	44.3 ± 0.5	12.5 ± 0.4	55.8 ± 0.3	48.2 ± 0.5	39.2
SagNet	57.7 ± 0.3	19.0 ± 0.2	45.3 ± 0.3	12.7 ± 0.5	58.1 ± 0.5	48.8 ± 0.2	40.3
MTL	57.9 ± 0.5	18.5 ± 0.4	46.0 ± 0.1	12.5 ± 0.1	59.5 ± 0.3	49.2 ± 0.1	40.6
ERM	58.1 ± 0.3	18.8 ± 0.3	46.7 ± 0.3	12.2 ± 0.4	59.6 ± 0.1	49.8 ± 0.4	40.9
MLDG	59.1 ± 0.2	19.1 ± 0.3	45.8 ± 0.7	13.4 ± 0.3	59.6 ± 0.2	50.2 ± 0.4	41.2
CORAL	59.2 ± 0.1	19.7 ± 0.2	46.6 ± 0.3	13.4 ± 0.4	59.8 ± 0.2	50.1 ± 0.6	41.5
MetaReg	59.8	25.6	50.2	11.5	64.6	50.1	43.6
DMG	65.2	22.2	50.0	15.7	59.6	49.0	43.6
ERM + FRR-L							
Default	63.6 ± 0.1	20.5 ± 0.0	50.7 ± 0.0	14.6 ± 0.1	63.8 ± 0.1	53.4 ± 0.0	44.2
RGD-1 **(Ours)**	65.8 ± 0.1	22.1 ± 0.0	52.3 ± 0.1	15.1 ± 0.1	65.7 ± 0.0	54.8 ± 0.1	**46.0**
RGD-EXP **(Ours)**	65.7 ± 0.1	21.9 ± 0.0	52.0 ± 0.1	15.1 ± 0.1	65.2 ± 0.1	54.9 ± 0.1	45.8
ERM + FRR							
Default	64.3 ± 0.1	21.2 ± 0.3	51.1 ± 0.2	14.9 ± 0.6	64.7 ± 0.1	54.1 ± 0.2	45.1
RGD-1 **(Ours)**	65.6 ± 0.0	21.5 ± 0.0	52.1 ± 0.0	15.0 ± 0.0	65.7 ± 0.0	55.1 ± 0.0	45.8
RGD-EXP **(Ours)**	65.6 ± 0.0	21.9 ± 0.0	52.0 ± 0.1	15.0 ± 0.1	65.5 ± 0.0	54.8 ± 0.1	45.8