Methods for Enhancing Robustness and Generalization in Machine Learning

by

Amit Schechter

B.S., Stanford University (2019)

Submitted to the Department of Electrical Engineering and Computer Science in Partial Fulfillment of the Requirements for the Degree of

Masters of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2024

Authored By: Amit Schechter
Department of Electrical Engineering and Computer Science
August 21, 2024

Certified By: Tommi S. Jaakkola
Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted By: Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee for Graduate Students

Methods for Enhancing Robustness and Generalization in Machine Learning

by

Amit Schechter

Submitted to the Department of Electrical Engineering and Computer Science
on August 21, 2024 in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

ABSTRACT

We propose two methods for improving subgroup robustness and out of distribution generalization of machine learning models. First we introduce a formulation of Group DRO with soft group assignment. This formulation can be applied to data with noisy or uncertain group labels, or when only a small subset of the training data has group labels. We propose a modified loss function, explain how to apply it to data with noisy group labels as well as data with missing or few group labels, and perform experiments to demonstrate its effectiveness. In the second part, we propose an invariant decision tree objective that aims to improve the robustness of tree-based models and address a common failure mode of existing methods for out-of-domain generalization. We demonstrate the benefits of this method both theoretically and empirically. Both these approaches are designed to enhance machine learning models' performance under distribution shift.

Thesis supervisor: Tommi S. Jaakkola
Title: Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to express my deepest gratitude to my advisor, Tommi Jaakkola, whose invaluable support made this work possible. His guidance, patience, and encouragement to explore my interests were instrumental throughout this project. Tommi's profound insight and technical expertise provided constant inspiration and direction, without which this work could not have been accomplished. His incredible personal support, understanding, and sensitivity were deeply appreciated.

I am also deeply grateful to my lab-mates and friends for their unwavering support and assistance. Their presence has created an environment that has fostered both my academic and personal growth, for which I feel incredibly fortunate.

Finally, I extend my heartfelt thanks to my family for their unconditional love and endless support, and to my dog Vince for providing the best emotional support and love throughout this journey.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Machine learning models achieve excellent performance on challenging learning tasks when the training and test data are drawn from the same distribution. However, they often fail in the presence of a distribution shift, limiting their applicability in many real-world scenarios [17, 47]. A major reason for the sensitivity of Empirical Risk Minimization (ERM) to minor data perturbations is that it tends to exploit non-causal correlations that are present in the training data [22]. These spurious features, being unstable and variable across environments, can lead to poor generalization.

Consider, for instance, a model trained to detect cows. If the training data predominantly features cows in grassy, green settings, the model might erroneously learn to rely more on the background than on the cow's features. Consequently, it may fail to identify cows in different contexts, such as on a beach [4]. This example illustrates how reliance on spurious correlations can compromise the model's robustness and generalization capabilities. Moreover, such a model might perform well on average, showing impressive overall performance on the dataset, while still failing on minority subgroups, such as cows in uncommon settings.

Our work addresses two closely related challenges: subgroup robustness and out-of-distribution (OOD) generalization. Subgroup robustness aims to improve the worst-group performance on the test set, while out-of-distribution generalization typically involves op-

timizing the worst-case performance over a perturbation set [58], where the choice of perturbation set reflects our assumptions about the test data. We assume the test distribution belongs to $\mathcal{E}_{\text{all}}$ and seek to minimize the worst-case risk over all potential test distributions, as shown in Equation (1.1). While we do not have direct access to the test distribution, we assume access to data from several training environments, which inform our understanding of the perturbation set.

$$\mathcal{R}^{OOD}(f) = \max_{e \in \mathcal{E}_{\text{all}}} \mathcal{R}^e(f) \tag{1.1}$$

Numerous approaches have been proposed to enhance the generalization capacity of neural networks, with most being *group-based*, enforcing the target metric over data subpopulations. Group distributionally robust optimization (Group DRO) [50] is one such method that minimizes the worst-group risk across all training groups. This approach is equivalent to minimizing Equation (1.1), where $\mathcal{E}_{\text{all}}$ is defined as mixtures of the training environments. Note: We use the terms *group* and *environment* interchangeably hereafter.

Group DRO is straightforward and performs well on various datasets. When groups are appropriately constructed, minimizing the worst-case group risk encourages the model to rely on stable features rather than spurious correlations, improving subgroup robustness and generalization. However, an important question arises: Can Group DRO enhance subgroup robustness when group information is noisy, missing, or unreliable? The first part of our work explores this question and proposes methods to improve subgroup robustness in the presence of noisy group information.

While methods like Group DRO can enhance out-of-distribution generalization in certain scenarios, their objectives do not explicitly avoid spurious correlations. Invariant learning approaches, such as Invariant Risk Minimization (IRM) [3], aim to directly address this issue. IRM learns relationships between inputs and targets that remain invariant across environments, creating representations with consistent optimal predictors across all environments. This approach is explicitly designed to eliminate spurious correlations in the learned

representation.

However, both IRM and Group DRO share a common failure mode: when the model can perfectly fit the training data, which is often the case with over-parameterized neural networks. In such instances, the IRM formulation degenerates to Empirical Risk Minimization (ERM), and Group DRO performs no better than ERM. The second part of our work builds upon this crucial observation.

Our work presents two main contributions. First, we introduce a modified Group DRO objective that employs soft, probabilistic group assignment instead of hard group assignment. This approach is particularly effective when dealing with noisy, missing, or unreliable group information. Second, we propose a robust decision tree model formulation. While extensive research has been conducted on out-of-distribution generalization for neural networks, few methods exist for decision trees. Traditional decision trees are prone to overfitting, and although ensemble methods like random forests and gradient boosting trees improve in-distribution generalization, they generally fall short in out-of-distribution scenarios. Our method explicitly encourages the tree to enforce invariance, thereby enhancing its generalization capabilities.

1.1 Overview of the Thesis

In Chapter 2 we provide an introduction to Group DRO and present our formulation of Weighted Group DRO using soft probabilistic group assignment.

In Chapter 3, we outline key use cases where our method is particularly beneficial. For each application, we detail the process of learning group assignments and present empirical evidence that demonstrates the effectiveness of our approach.

In Chapter 4 we propose a formulation for robust decision trees, discuss the benefits of invariance in decision trees, and demonstrate effectiveness empirically.

Finally, Chapter 5 summarizes our work and discusses future research directions.

Chapter 2

Weighted Group DRO

2.1 Background

Many methods aim to improve robustness and generalization by training models that avoid reliance on spurious correlations. Group Distributionally Robust Optimization (Group DRO) [50] is one such approach, enhancing subgroup robustness by minimizing the worst-case group risk across all training groups. When groups are appropriately constructed, this minimization encourages the model to rely on stable features rather than spurious correlations. In practice, groups are often formed based on combinations of attributes and labels. For instance, in the Waterbirds dataset [50], water backgrounds strongly correlate with the "waterbird" label. An Empirical Risk Minimization (ERM) model aimed to classify the type of bird might heavily rely on this background to achieve low average risk. However, by constructing groups where this correlation doesn't hold (e.g., a group with many waterbirds on land backgrounds), and minimizing the worst-group risk rather than the average risk, the model is more likely to depend on features beyond the background. Notably, Group DRO is a specific instance of distributionally robust optimization (DRO) [8, 16], which optimizes for the worst-case loss over a set of potential test distributions. In Group DRO, this uncertainty set is defined as a mixture of training groups to minimize the worst-group risk.

Group DRO, like most methods for out-of-distribution generalization and robustness, typically assumes that each data sample belongs to exactly one group and that we have complete knowledge of group assignments in the training data. This assumption enables optimization of the worst-group loss. However, in practice, we might only have access to noisy group labels. There are numerous reasons for such noisy or unreliable group labels. For instance, in the Waterbirds dataset, group affiliation might be ambiguous in cases where an image background contains both water and land (see Figure 2.1). Moreover, attributes might be automatically labeled, potentially introducing errors, or manually annotated by humans, which could also lead to incorrect classifications. These examples merely scratch the surface of the abundant sources for noisy and uncertain group annotations.

Naively applying Group DRO when group assignments are noisy may fail to effectively optimize the worst-group risk and address spurious correlations as intended. Consider a scenario where some images of waterbirds on water backgrounds are mislabeled as waterbirds on land. Optimizing the Group DRO objective under these noisy labels might lead us to believe we're improving performance on the challenging "waterbirds on land" group, when in reality, we're merely reinforcing performance on the more common "waterbirds on water" group. In the extreme case of entirely random group labels, they cease to provide any meaningful information about spurious correlations, likely resulting in significantly deteriorated worst-group performance.

Furthermore, in cases of missing or ambiguous group assignments, using binary labels can be inaccurate and lead to less effective data groupings. Instead, we can leverage this uncertainty to construct more nuanced and effective groups. This can be achieved by assigning varying weights to examples based on how representative they are of each group. For instance, images of waterbirds with purely water backgrounds should contribute more significantly to the "waterbird on water" group than those with mixed water and land backgrounds. By employing this "soft group assignment" approach, we can make more effective use of the available information.

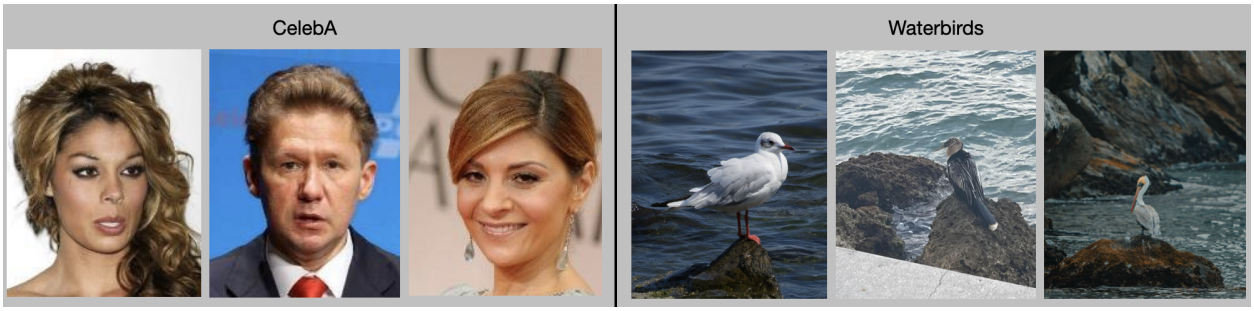


Figure 2.1: Examples of ambiguous attributes and uncertain groups. On the left, images from the CelebA dataset illustrate cases where hair color is not clearly blonde or dark. On the right, examples from the Waterbirds dataset show ambiguous backgrounds that include a combination of water and land.

Our approach also draws inspiration from research on robust optimization for fairness in scenarios with noisy protected group labels. One notable method [59] focuses on bounding fairness violations for true groups when fairness criteria are satisfied on noisy groups. The authors propose a formulation that guarantees the fairness criteria are met for the true protected groups, even when working with noisy data. While we operate in a similar setting, our objective differs: instead of enforcing strict fairness criteria, we aim to enhance subgroup robustness.

In this work, we propose an algorithm that improves Group DRO performance under conditions of noisy, missing, or uncertain group assignments. Rather than assigning each data sample to a single group, we define group assignments as a distribution over all possible groups and incorporate this probabilistic soft group assignment into the Group DRO objective. We introduce a method to optimize this objective and conduct experiments to demonstrate its effectiveness.

2.2 Related Work

Domain Generalization: In recent years, numerous studies have aimed to improve out-of-distribution generalization, with a particular focus on domain generalization. When group annotations are available in the training dataset, Group Distributionally Robust Optimiza-

tion [50] can enhance subgroup robustness and out-of-distribution generalization by minimizing the worst-group loss. Other approaches involve learning invariant features [3] or employing adversarial training techniques [20] to learn features that are indistinguishable across domains. In cases where only partial group annotations are available, useful methods include Deep Feature Reweighing [32], which retrains the last layer on a group-balanced held-out set, and Just Train Twice [37], which fine-tunes the model by upweighting misclassified examples. When no group information is available, some methods attempt to estimate group assignments by identifying misclassified examples using a proxy model. Finally, a study by [62] addresses a scenario related to ours by improving domain generalization under noisy group information through the use of environment label smoothing in domain adversarial training.

Fairness with Imperfect Group Annotations: Most methods that improve subgroup robustness assume accurate group information, but some work in the field of fairness accounts for noisy or missing group annotations. One approach assumes noisy group labels and proposes methods for enforcing fairness criteria using DRO-based approaches [59]. Another method assumes no group information and uses robust optimization to minimize the worst-case risk over all distributions close to the empirical distribution [26]. While these methods address settings with partial or noisy group information, they focus on satisfying strict fairness criteria rather than optimizing an objective aimed at improving performance across all groups.

Learning with Noisy and Uncertain Labels: While few works consider noisy group annotations, a significant amount of research has focused on improving learning with noisy labels [53]. Some methods demonstrate the effectiveness of label smoothing in addressing noisy labels [41]. Others modify the model architecture, aiming to learn the noise transition matrix of the noisy data or incorporating regularization techniques to mitigate the effects of noise [60]. Additional approaches include estimating the true labels using latent variable models [5]. Modified loss functions are also commonly used to address label noise [52]. These

approaches include estimating the transition matrix and integrating it into the loss function, reweighting data samples, and modifying the loss to combine both noisy and predicted labels. Methods that address label uncertainty often focus on identifying the more certain samples in the data and using self-supervised approaches to iteratively improve predictions [48], but these methods can be limited when the model is not well-calibrated. While work on label noise and uncertainty provides useful inspiration and background, our work addresses a different setting where the role of the group label is distinct from that of the predicted label.

To the best of our knowledge, only one study specifically aims to improve generalization under noisy group annotations [62]. This work uses environment label smoothing in the context of adversarial domain generalization.

2.3 Problem Setup

Consider the task of predicting labels $y \in \mathcal{Y}$ from input features $x \in \mathcal{X}$, where the training data is drawn from some distribution P_{train}. The standard goal of Empirical Risk Minimization is to fit a model θ that minimizes the expected loss $\mathbb{E}_{P_{\text{train}}}[\ell(x, y; \theta)]$, where the training and test data are assumed to be sampled from the same distribution. In subgroup robustness, we aim to minimize the risk of the worst-performing subgroup. We assume access to training data $\mathcal{D}_{\text{train}} \sim P_{\text{train}}$, where the training distribution P_{train} is a mixture of G groups, and $(x, y) \sim P_g$ denotes data sampled from group g. We operate within the framework of subgroup robustness, utilizing labeled data from training groups only. However, our method can also be applied in other scenarios where Group DRO is applicable.

$$\min_{\theta} \max_{g \in \mathcal{G}} \mathbb{E}_{(x,y)\sim P_g}[\ell(x, y; \theta)]$$

If we know which group each training point comes from, i.e., we have access to (x, y, g) for the training data, we could optimize the worst-group risk across the training data as

shown in the equation above (2.3). However, we work under the assumption that the group labels are noisy or missing. Thus, optimizing this equation directly is not possible, and we propose alternative formulations.

Noisy Group Assignment: Unlike most existing work on subgroup robustness and generalization, we assume that we observe a noisy version of the group assignment rather than the true group assignment. We denote this noisy group assignment by $\tilde{g} \in \mathcal{G}$. Thus, we have access to training data of the form $\tilde{D}_{\text{train}} = \{(x_i, y_i, \tilde{g}_i)\}_{i=1}^{N}$, where $P_{\tilde{g}}$ represents the distribution of data samples from the noisy group $\tilde{g}$. We have a noise model that describes the relationship between the noisy label, the true label, and the data: $P(\tilde{g}|x, y, g)$, which we elaborate on in Section 3.1.1. The relationship between the true joint distribution and the observed distribution can be formulated as $P(x, y, \tilde{g}) = \sum_g P(x, y, \tilde{g}, g) = \sum_g P(x, y, g)P(\tilde{g}|x, y, g)$. In this settings we will estimate $P(g|x, y, \tilde{g})$ and use it in our formulation.

Missing Group Information: In this setting, we have access to a large training set without group information, as well as a small set of examples with their corresponding true group assignments. We will use this small dataset to learn $P(g|x, y)$ and incorporate it into our soft Group DRO formulation. If the model predicting $P(g|x, y)$ is well-calibrated, we can expect performance improvement by assigning higher weights to data samples that more certainly belong to a specific group.

Both scenarios described above can be addressed using a probabilistic version of Group DRO, where group assignment is treated as a soft assignment rather than a binary hard assignment. We propose an algorithm to handle these scenarios, highlighting the differences between the two cases when necessary. In Chapter 3, we provide a more detailed explanation of the setup for each scenario.

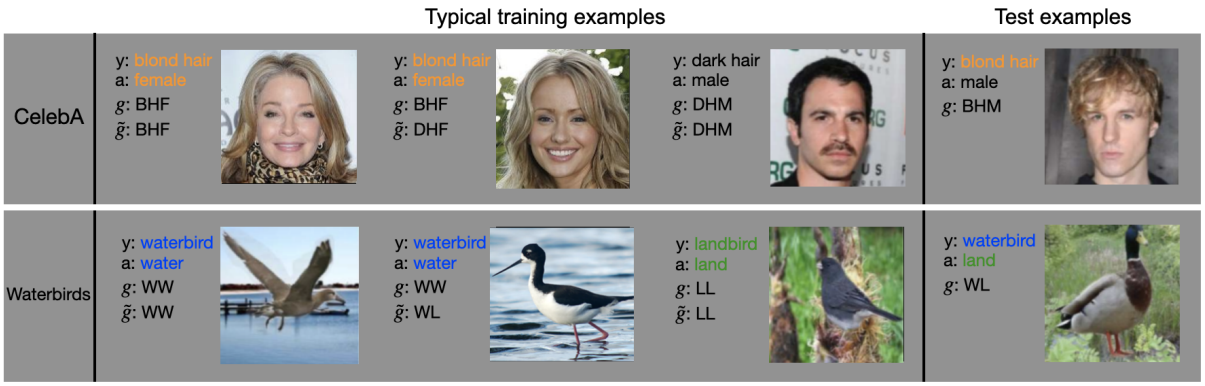


Figure 2.2: Representative training and test examples for the datasets we consider. During training, we observe only the noisy group assignment $\tilde{g}$. The correlation between the label y and the spurious attribute a observed in the training data does not hold at test time.

2.4 Method

This work is an instance of distributionally robust optimization (DRO) [8, 16], which optimizes the worst-case loss over a set of distributions (2.1). The general form of DRO aims to minimize the worst-case loss over an uncertainty set of distributions $\mathcal{Q}$:

$$\min_{\theta} \left\{ \mathcal{R}(\theta) := \sup_{Q \in \mathcal{Q}} \mathbb{E}_{(x,y)\sim Q} \left[\ell(x, y; \theta) \right] \right\} \tag{2.1}$$

The uncertainty set $\mathcal{Q}$ encodes the possible test distributions on which our model should perform well. Group DRO [50] is a specific instance of DRO which defines the uncertainty set $\mathcal{Q}$ as the set of any mixture of groups in the training data, thus optimizing the expected loss of the worst-case group:

$$\min_{\theta} \left\{ \max_{g \in \mathcal{G}} \mathbb{E}_{(x,y)\sim P_g} \left[\ell(x, y; \theta) \right] \right\} \tag{2.2}$$

Traditional Group DRO assumes access to correct group labels, with each sample belonging to exactly one group. However, this approach is not optimal for our setting, where group membership is noisy or uncertain. In such cases, group assignment is better described as

probabilistic rather than a hard binary assignment. Our method is formulated to address this issue, extending the Group DRO framework to accommodate uncertain group memberships.

2.4.1 Weighted Group DRO using Soft Group Assignment

We can rewrite the objective of Group DRO as a minimax formulation over the average loss for all data samples belonging to group g:

$$\min_{\theta} \max_{g \in \mathcal{G}} \frac{1}{N_g} \sum_{i=1}^{N} \mathbb{1}(g_i = g) \ell(x_i, y_i; \theta)$$

Here, N_g is the number of samples in group g and $\ell(x_i, y_i; \theta)$ is the loss for sample i.

In our formulation, we use probabilistic group assignment rather than hard group assignment. It's natural to replace $\mathbb{1}(g_i = g)$, which indicates whether sample i belongs to group g, with a term expressing the probability that sample i belongs to group g.

We define $q_i(g) = P(g|x_i, y_i, \tilde{g}_i)$, where $\tilde{g}$ represents noisy group labels. In scenarios where we don't have access to noisy group labels, this becomes $P(g|x_i, y_i)$. We define N_g according to the probabilistic assignment: $N_g = \sum_{i=1}^{N} q_i(g)$. The objective becomes:

$$\min_{\theta} \max_{g \in \mathcal{G}} \left[\frac{1}{N_g} \sum_{i=1}^{N} q_i(g) \cdot \ell(x_i, y_i; \theta) \right] \tag{2.3}$$

$$= \min_{\theta} \max_{g \in \mathcal{G}} \left[\sum_{i=1}^{N} w_i(g) \cdot \ell(x_i, y_i; \theta) \right] \text{ where } w_i(g) = \frac{q_i(g)}{\sum_{j=1}^{N} q_j(g)} \tag{2.4}$$

Note that $w_i(g)$ is the normalized weight, i.e., the amount that point i contributes to the loss of group g. If the true group label of data point i is likely to be g, i.e., $P(g|x_i, y_i, \tilde{g})$ is high, then $w_i(g)$ is high. However, the weight is also affected by N_g, so if many examples are likely to belong to group g, the contribution of each one of them is smaller. In the case where all the group probabilities are binary, the weights $w_i(g)$ are equal to $\frac{1}{N_g}$ if example i belongs to group g and 0 otherwise, and the objective becomes the regular Group DRO

objective.

If we had access to the group probabilities $p(g|x_i, y_i)$ for all i, g, we could directly optimize this objective to find θ, similar to Group DRO but with different weights. This becomes a general formulation for incorporating probabilistic group assignment into the Group DRO objective. Algorithm 1 describes the learning process using the Weighted Group DRO objective.

Since we don't have direct access to the probabilities $q_i(g)$, we need to estimate them. Once we have these estimated probabilities, we can use them in the Weighted Group DRO objective. In the next section, we describe methods for estimating $q_i(g)$ for both scenarios with noisy group labels and those with few or uncertain group labels.

Algorithm 1 Group DRO with Soft Group Assignment

Input: Training data $\{(x_i, y_i)\}_{i=1}^n$
Input: Soft group assignment $q_i(g)$ for all i, g

Initialize model parameters θ
for each epoch **do**
    for each mini-batch B **do**
        Compute probabilistic group sizes $N(g) = \sum_{i \in B} q_i(g)$ for all $g \in G$
        Compute loss for each training example $\ell_i(x_i, y_i; \theta)$ for all $i \in B$
        Compute group losses $\ell(g) = \frac{1}{N(g)} \sum_{i \in B} q_i(g) \ell_i(x_i, y_i; \theta)$ for all $g \in G$
        $\theta \leftarrow \arg\min_\theta \max_{g \in \mathcal{G}} \ell(g)$ ▷ Update parameters to minimize the max group loss
    end for
end for

Chapter 3

Applications of Weighted Group DRO

Our proposed method is applicable in scenarios where Group Distributionally Robust Optimization (DRO) can be used and group probabilities can be estimated. We focus primarily on cases where noisy group labels are available during training, but true group labels are not available. For this scenario, we propose a method for estimating group probabilities and empirically demonstrate its effectiveness.

We then extend our approach to situations where we have a small set of accurately labeled group information, and no noisy group annotations. In both scenarios, we employ different techniques to estimate the group assignments, after which we apply our method using the group probabilities $p(g|x, y)$.

3.1 Noisy Group Labels

In this section, we examine the scenario where our training data contains noisy group labels. The dataset can be represented as $\{(x_i, y_i, \tilde{g}_i)\}$, where x_i represents the input features, y_i denotes the target variable, and $\tilde{g}_i$ indicates the noisy group label. It's important to note that during the testing phase, we evaluate the worst-group performance using the true group labels g, not the noisy ones. We will address the potential pitfalls of naively applying Group DRO when working with noisy group labels. Additionally, we will demonstrate how our

proposed Weighted Group DRO formulation can be leveraged to enhance performance in the presence of label noise.

3.1.1 Noise modeling

We begin by examining various noise models responsible for generating noisy group labels.

Instance-independent noise: the observed group label is conditionally independent of the data given the true group assignment, i.e. $P(\tilde{g}|x, y, g) = P(\tilde{g}|g)$. The relationship between the observed data and the true group label can be written as:

$$P(x, y, \tilde{g}) = \sum_{g} P(x, y, g)P(\tilde{g}|g) \tag{3.1}$$

A commonly adopted noise model is symmetric noise, where $P(\tilde{g}|g)$ is the same for all $\tilde{g} \neq g$.

Instance-dependent noise: in this scenario, $P(\tilde{g}|x, y, g) \neq P(\tilde{g}|g)$, indicating that the noise is data-dependent. This necessitates a more complex model to approximate $P(\tilde{g}|x, y, g)$. In this case the noise may vary based on the data. For example, in the waterbirds case, land background with a significant portion of blue (e.g., sky) is more likely to be mislabeled as having a water background.

3.1.2 Performance gap

Given training data $\{x_i, y_i, \tilde{g}_i\}_{i=1}^{N}$, one could naively apply Group DRO to this problem by using the noisy groups as group assignments.

$$\min_{\theta} \left\{ \max_{g \in \mathcal{G}} \mathbb{E}_{(x,y)\sim \tilde{P}_g} \left[\ell(x, y; \theta) \right] \right\}$$

However, Group DRO assumes that the group labels are accurate and does not account for the possibility of group label noise. Applying Group DRO in such cases leads to poor performance. In the extreme case where the noisy group labels are sampled uniformly at

random from all possible labels, we expect a significant drop in worst-group performance. This is because the random assignment renders the group labels meaningless, effectively destroying any meaningful group structure in the data. Empirically, we demonstrate the performance degradation that occurs when Group DRO is applied with noisy group labels compared to using the true group labels (see Table 3.1).

Experiments

We compare the performance of ERM, Group DRO, and Weighted ERM in the presence of noisy group annotations on Waterbirds [50] and CelebA [38]. Group DRO is the model proposed by [50]. Weighted ERM, which was used as a baseline in [50], is trained using the ERM objective on data sampled from each group with equal probability. All models are pre-trained ResNet50 models, fine-tuned on our task. Since ERM does not use the group labels, we only train it once. We train both the Group DRO and Weighted ERM models twice: once using the true group labels and once using noisy group labels. We then present the average and worst-group test accuracy. For the noisy group scenario, we generate the noisy groups by randomly reassigning 10% of the training examples a new group label, where the new group label is sampled uniformly at random from the remaining $G - 1$ groups.

The models are trained similarly to the original Group DRO paper. We fine-tune a pre-trained ResNet 50 model. We use learning rate of 0.001 for Waterbirds and 0.0001 for CelebA. We train the models for 10 epochs on CelebA and 20 epochs on Waterbirds, and select the model with the highest worst-group validation accuracy. The results, presented in Table 3.1, reveal a substantial decline in performance for both the Group DRO and Weighted ERM models when trained with noisy group labels. All three models exhibit poor performance in the presence of group noise. This significant performance gap motivates our proposal of a method that is more robust to group noise.

Dataset	Group Type	Avg Accuracy			Worst Accuracy		
		ERM	WERM	GDRO	ERM	WERM	GDRO
CelebA	True Groups	95.7%	91.5%	91.6%	40.0%	83.3%	**85.0%**
	Noisy Groups	95.7%	94.0%	93.6%	40.0%	74.4%	**79.4%**↓6%
Waterbirds	True Groups	84.5%	91.7%	90.9%	67.1%	85.4%	**86.5%**
	Noisy Groups	84.5%	92.5%	89.6%	67.1%	72.7%	**77.9%** ↓9%

Table 3.1: Comparison of ERM, Weighted ERM, and Group DRO models across multiple datasets, trained with true vs. noisy group labels. Noisy group labels are generated by randomly reassigning 10% of the true group labels. Performance drop when using Group DRO with noisy group labels are highlighted in red, showing percentage decreases compared to the same model trained with the true group labels.

3.1.3 Soft Group Assignment

In this section, we discuss the application of our method in cases involving noisy group labels. Algorithm 1 outlines the optimization process, where we first determine the group assignment $q_i(g) = P(g|x_i, y_i, \tilde{g}_i)$ and then minimize the Weighted Group DRO objective based on $q_i(g)$. To estimate the soft group probabilities, we assume access to the training data $\{(x_i, y_i, \tilde{g}_i)\}_{i=1}^n$, where x_i represents the input features, y_i the true label, and $\tilde{g}_i$ the potentially noisy group label. We propose two methods for estimating the group assignment $q_i(g)$.

- **Group Label Smoothing:** The first and simplest approach is to apply group label smoothing to the noisy group labels. This method offers a straightforward way to incorporate the noise model into the objective without introducing additional computational complexity. Previous work, such as [41], has demonstrated the effectiveness of label smoothing in addressing label noise. For categorical group labels, we use uniform smoothing:

$$q_i(g) = P(g|\tilde{g}_i) = \begin{cases} 1-\alpha, & \text{if } g = \tilde{g}_i \\ \frac{\alpha}{G-1}, & \text{otherwise} \end{cases} \tag{3.2}$$

Here, α controls the amount of smoothing ($\alpha = 0$ means no smoothing).

- **Parameterized model**: rather than determining the group g solely based on the noisy group $\tilde{g}$, we can learn to estimate it from the data using a parameterized model. The relationship between the group assignment and the observed data can be expressed as:

$$P(g|x, y, \tilde{g}) = P(\tilde{g}|x, y, g)\frac{P(g|x, y)}{P(\tilde{g}|x, y)} \propto P(\tilde{g}|x, y, g)P(g|x, y)$$

Assuming access to the noise model $P(\tilde{g}|x, y, g)$, e.g. a simple perturbation matrix $P(\tilde{g}|g)$, we only need to estimate $P(g|x, y)$. Since g is latent, we will use an Expectation-Maximization (EM) approach to estimate it, as described below.

EM Algorithm Details:

Our goal is to train a classifier to predict the true group label given the observed data. In Figure 3.1, we describe the modeling process, where $(x, y, \tilde{g})$ are observed and g is latent. The neural network mapping (x, y) to g is parameterized by θ, learning to model $P(g|x, y; \theta)$. The noise model can be given in advance or parameterized, but for simplicity, we assume access to the noise model $P(\tilde{g}|g)$, which is given as a $G \times G$ matrix.

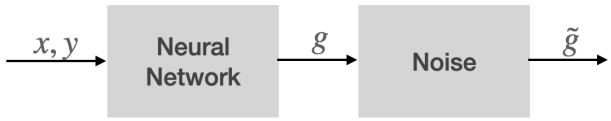


Figure 3.1: Data relationship: x, y and $\tilde{g}$ are observed, g is latent.

During training, we are given n training samples $\{(x_i, y_i, \tilde{g}_i)\}_{i=1}^n$. The log-likelihood of the data given the model parameters θ can be written as:

$$L(\theta) = \sum_{i=1}^{n} \log p(\tilde{g}_i|x_i, y_i; \theta) = \sum_{i=1}^{n} \log \left(\sum_{g \in G} p(\tilde{g}_i|g)p(g|x_i, y_i; \theta) \right) \tag{3.3}$$

Our goal is to use the training data to learn the optimal parameters θ that maximize the likelihood of the data. Since the group label g is latent, we use the Expectation-Maximization (EM) algorithm to optimize θ, and then use the model to predict $p(g|x_i, y_i; \theta)$. Since the objective above cannot be directly optimized, we follow a standard approach in Expectation-Maximization using the Evidence Lower Bound (ELBO). For simplicity, we derive the objective for a single data point, and then reintroduce the summation over the data.

$$\log p(\tilde{g}|x, y; \theta) = \log \left[\sum_{g \in G} p(g|x, y; \theta) p(\tilde{g}|g) \right] \tag{3.4}$$

$$= \log \left[\sum_{g \in G} Q(g|x, y, \tilde{g}) \frac{p(g|x, y; \theta) p(\tilde{g}|g)}{Q(g|x, y, \tilde{g})} \right] \tag{3.5}$$

$$\geq \sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[\frac{p(g|x, y; \theta) p(\tilde{g}|g)}{Q(g|x, y, \tilde{g})} \right] \tag{3.6}$$

$$= \sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[p(g|x, y; \theta) p(\tilde{g}|g) \right] + \sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[\frac{1}{Q(g|x, y, \tilde{g})} \right] \tag{3.7}$$

The second term does not depend on θ and we can drop it when we optimize θ.

$$\sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[p(g|x, y; \theta) p(\tilde{g}|g) \right]$$

Since $p(\tilde{g}|g)$ remains constant, we can drop it and optimize the objective:

$$\sum_{g \in G} Q(g|x, y, \tilde{g}) \log p(g|x, y; \theta)$$

We define $Q(g|x, y, \tilde{g})$ as our soft estimate of the group probabilities, and $p(g|x, y; \theta)$ is the model prediction using the most recent parameters θ. The EM algorithm steps

are shown in Algorithm 2.

Algorithm 2 EM Algorithm for modeling $P(g|x,y)$

Input: Training data $\{(x_i, y_i, \tilde{g}_i)\}_{i=1}^n$
Output: Group probabilities $P(g|x,y)$
Initialization:
Initialize the noise model $P(\tilde{g}|g)$ to a noise transition matrix of size $G \times G$
Repeat:
    E-step: calculate the group probabilities for all i, g:

$$q_i(g) = p(g|x_i, y_i, \tilde{g}_i; \theta) \propto p(g|x_i, y_i; \theta) p(\tilde{g}_i|g)$$

    M-step: update θ to maximize the following objective:

$$\sum_{i=1}^{n} \sum_{g \in G} q_i(g) \log p(g|x_i, y_i; \theta)$$

until Convergence
Return: Group probabilities $p(g|x,y)$

3.1.4 Experiments

To study the behavior of Weighted Group DRO in the presence of noisy group annotations, we train ResNet50 models [27] on the CelebA [38] and Waterbirds [50] datasets. We train models using ERM, Weighted ERM, Group DRO, and Weighted Group DRO under conditions where the training and validation data contain noisy group annotations, while the test data has the true group annotations.

Data

Waterbirds dataset [50]: This dataset contains images of waterbirds and landbirds on water background and land background. The label is the type of bird $\mathcal{Y} = \{\text{waterbird, landbird}\}$ and the attribute is the background $\mathcal{A} = \{\text{water background, land background}\}$. The background spuriously correlates with the label, with waterbirds more frequently appearing on water background, and landbirds frequently appearing on land background (see Appendix C for data statistics). There are $n = 4,795$ training examples, with the smallest group

(waterbirds on land background) containing only 56 training examples.

CelebA dataset [38]: Dataset containing images of celebrities along with attributes such as hair color and gender. We study the ability of models to achieve good worst-group performance in the presence of spurious correlations between the label (hair color $\mathcal{Y} = \{\text{blond, dark}\}$) and the spurious attribute (gender $\mathcal{A} = \{\text{male, female}\}$). There are $n = 162,770$ training examples, with $1,387$ training examples in the smallest group (male, blond hair).

Group Noise Generation: We generate the noisy group annotations $\tilde{g}$ by introducing noise to the true group label g as follows:

- **Uniform noise:** We use the following process:
  - With probability $1 - p$, we keep the true group label unchanged: $\tilde{g} = g$.
  - With probability p, we randomly assign a new label, sampling one of the other $G - 1$ group labels with equal probability.

  We repeat this process adding uniform noise with $p = 0.1$ and $p = 0.3$.

- **Variable noise:** We use the following process, using $p = 0.1$:
  - With probability $1 - p$, we keep the true group label unchanged: $\tilde{g} = g$
  - With probability p, we assign a new group label using according to a pre-defined permutation set that varies based on the true group label and is described in detail in Appendix A.

Models

The classification models are all pre-trained Resnet50 models. We use similar hyperparameters to those in the original Group DRO paper [50]. The models are trained using stochastic gradient descent with a momentum term of 0.9 and batch size of 128. As in the original paper, we use batch normalization and no dropout. We use learning rate of 0.0001 for CelebA

and 0.001 for Waterbirds, following previous methods [50]. For simplicity, we train all models without data augmentation. We train the model for 10 epochs for CelebA and 20 epochs for the Waterbirds data, and choose the model with the highest worst-group validation accuracy. ERM is only trained once for each dataset, as it does not utilize group annotations. Weighted ERM, Group DRO, and Weighted Group DRO are trained separately for each group label noise scenario. They are trained similarly to the previous implementation proposed in [50], but Weighted Group DRO uses our modified objective (2.4). Complete implementation details in Appendix A. For each model, we present the worst-group accuracy measured on the test set, using the true group annotations.

Group Probability Model

The Weighted Group DRO model optimizes objective 2.4, as described in Algorithm 1. We consider two approaches for determining the group probabilities which are later used by the Weighted Group DRO algorithm:

- **WGDRO Smoothed:** We use uniform group label smoothing. This approach sets the group probabilities to be a smoothed version of the noisy group label:

$$P(g|x_i, y_i\tilde{g}_i) = \begin{cases} 1-\alpha & \text{if } g = \tilde{g}_i \\ \frac{\alpha}{G-1} & \text{otherwise} \end{cases}$$

  We experimented with smoothing parameters $\alpha \in \{0.05, 0.1, 0.15\}$. For our final experiments, we used the lowest value tested, $\alpha = 0.05$, as it achieved the highest worst-group validation accuracy.

- **WGDRO Expectation–Maximization (EM):** The group probabilities are estimated using the EM algorithm 2. The group prediction model $p(g|x, y; \theta)$ is a convolutional neural network (CNN) with the following architecture: Three convolutional layers with ReLU activation and max pooling, followed by adaptive average pooling, and two fully connected layers. We use Adam optimizer with a learning rate of 0.001,

Dataset	Method	Group Label Noise			
		No Noise	Uniform, 0.1	Var, 0.1	Uniform, 0.3
CelebA	ERM	40.0%	40.0%	40.0%	40.0%
	WERM	83.3%	74.4%	76.7%	58.9%
	GDRO	**85.0**%	79.4%	78.3%	63.3%
	WGDRO-S	85.0%	76.7%	78.3%	57.8%
	WGDRO-EM	83.3%	**90.0**%	**81.7**%	**82.8**%
Waterbirds	ERM	67.1%	67.1%	67.1%	67.1%
	WERM	85.4%	72.7%	74.8%	72.0%
	GDRO	**86.5**%	77.9%	75.5%	69.9%
	WGDRO-S	83.6%	74.3%	76.0%	67.9%
	WGDRO-EM	80.7%	**81.2**%	**80.2**%	**74.6**%

Table 3.2: Worst-group performance on CelebA and Waterbirds with different noise levels. Rows highlighted in gray indicate our proposed method WGDRO-EM.

which we selected after experimenting with different learning rates $(1e^{-2}, 1e^{-3}, 1e^{-4})$ and monitoring the loss. The batch size is set to 128. We trained the model for 20 epochs for CelebA and 40 epochs for Waterbirds, which was determined by monitoring the loss, the likelihood $p(\tilde{g}|x, y; \theta)$, and evaluating the validation performance of the downstream label classification model. For the noise model, we use a $G \times G$ matrix with the following properties: $(1 - \alpha)$ on the diagonal, and α everywhere else. We experimented with three different noise model parameters: $\alpha = \{0.05, 0.1, 0.15\}$. For each noise level, we trained the EM algorithm and the Weighted Group DRO model on top of it and recorded the worst-group validation accuracy of the label prediction task. We selected the best performing noise model of $\alpha = 0.1$. See complete implementation details in the Appendix A. For simplicity, we used a symmetric noise model. However, one could experiment with alternative noise models, or the noise model could be parameterized and learned jointly in the EM algorithm, following a similar approach to [5]. Once the group prediction model is trained, we evaluate its performance by comparing the predicted group probabilities to the true groups, calculating per-group precision and recall. These results are recorded in Appendix C.

Results

Our study demonstrates that our method, Weighted Group DRO, significantly improves subgroup robustness when group labels are noisy. Table 3.2 presents the results of different methods on CelebA and Waterbirds datasets under varying levels of group label noise. These include no noise, uniform noise with probabilities of 0.1 and 0.3, and variable noise with a probability of 0.1 that is group-dependent, increasing the likelihood of specific perturbations. These diverse noise conditions allow us to evaluate the robustness of each method across different challenging scenarios.

As seen in Table 3.2, while the original Group DRO formulation outperforms Weighted Group DRO in the absence of group label noise, WGDRO-EM shows superior performance in all cases when noise is present. The performance of Weighted Group DRO is strongly correlated with the accuracy of the predicted group probabilities. Weighted Group DRO with smoothed probabilities (WGDRO-S) shows poor performance in most scenarios, indicating that simple smoothing is insufficient. In contrast, Weighted Group DRO with EM-learned probabilities (WGDRO-EM) outperforms all other models in the presence of group noise, suggesting successful learning of group probabilities. For CelebA, worst-case test accuracy improves significantly, with WGDRO-EM outperforming Group DRO by 3.4%, 10.6%, and 19.5% for variable noise ($p = 0.1$), uniform noise ($p = 0.1$), and uniform noise ($p = 0.3$), respectively.

These results demonstrate that Weighted Group DRO (WGDRO) offers a promising approach for improving predictions in the presence of noisy group annotations. As noise levels increase, standard Group DRO performance deteriorates significantly, while WGDRO effectively mitigates this effect. Our method's robustness to group label noise suggests its potential applicability in real-world scenarios where perfect group annotations are often unavailable. Future work could explore the application of WGDRO to other domains and investigate its effectiveness with different types of group noise.

Analysis

To analyze the relationship between the quality of group probability estimates and the performance of the prediction model, we conduct a series of experiments with multiple group prediction EM models. We select models with varying precision and recall performance to observe how these metrics affect the downstream classification task. Our process is as follows:

1. We train multiple EM models to estimate group probabilities, using the same architecture described earlier with differing hyperparameters.

2. For each EM model, we record its worst-group training precision and recall.

3. Using the probabilities predicted by each EM model, we train a corresponding classification model (Weighted Group DRO) with the same architecture as in previous experiments.

4. We evaluate the worst-case test accuracy of each WGDRO classification model.

Table 3.3 presents these results, showing the worst-group precision and recall of various group prediction models alongside the corresponding worst-group test accuracy of their WGDRO-EM classification models.

Notably, we observe that the WGDRO-EM model can achieve high worst-group test accuracy even when the EM model doesn't assign the minority group as the most likely group for any training examples. This is a significant advantage over standard Group DRO, where such a minority group would be entirely overlooked during optimization. Our use of soft group assignments allows us to improve performance across all groups, including minority ones.

3.2 Few or Missing Group Labels

We describe how our method can be adapted for scenarios with limited or missing group annotations. In this setting, we assume access to two training datasets:

Precision	Recall	Pred Acc
72.7%	76.7%	84.7%
55.3%	73.3%	82.4%
79.0%	60.7%	85.0%
79.1%	21.4%	81.0%
86.0%	19.6%	81.3%
0.0%	0.0%	83.7%

Table 3.3: Performance comparison for Waterbirds data with uniform noise = 0.1. Precision and recall are measured for the group prediction model on the worst-group in the training data. Accuracy is measured on the label classification model for the worst-case test group.

Noise	Precision	Recall	Pred Acc
Uniform 0.1	93.1%	93.3%	90.8%
Uniform 0.1	75.2%	92.8%	85.0%
Uniform 0.1	84.7%	47.0%	88.3%
Variable 0.1	95.2%	89.3%	88.3%
Variable 0.1	94.8%	64.6%	85.6%
Variable 0.1	85.9%	50.2%	84.4%

Table 3.4: Performance comparison for CelebA data with uniform noise = 0.1 and variable noise = 0.1. Precision and recall are measured for the group prediction model on the worst-group in the training data. Accuracy is measured on the label classification model for the worst-case test group.

- A large unlabeled dataset $D_u = \{(x_i, y_i)\}_{i=1}^{n_u}$, where group annotations are not available.
- A small labeled dataset $D_l = \{(x_i, y_i, g_i)\}_{i=1}^{n_l}$, where group annotations are present.

Typically, $n_l \ll n_u$, reflecting the common scenario where labeled data is scarce compared to unlabeled data. In our setting, only 10% of the training data includes group annotations. These annotations are sampled in a stratified manner to ensure sufficient representation of minority groups (details in Appendix A). No noise is added to the group labels in this case.

To apply our proposed Weighted Group DRO formulation in this context, we need to estimate $P(g|x, y)$ for all training examples. We use the following approach:

- Train a simple neural model on the labeled data D_l to estimate the group probabilities $P(g|x, y; \theta)$, call this model f_θ.
- Use f_θ to estimate $P(g|x, y; \theta)$ for the unlabeled examples in D_u.

This approach allows us to extend the benefits of our method to situations where complete group annotations are unavailable or impractical to obtain for the entire dataset.

3.2.1 Experiments

As described above, we train f_θ to estimate $P(g|x_i, y_i; \theta)$. We use a simple Convolutional Neural Network [35] with two convolutional layers with ReLU activation and max pooling, and a fully connected layer, trained with Adam optimizer using learning rate 0.001 and batch size 128 (complete implementation details in Appendix A). We then use f_θ to calculate the group probabilities $P(g|x, y; \theta)$ for the unlabeled examples D_u.

We compare the performance of an ERM model which doesn't require group annotations, Weighted Group DRO which uses the predicted probabilities using f_θ, and Group DRO model using the same predicted probabilities probabilities using f_θ, converted into binary group assignments by assigning the highest-probability group as the group label. We record the average test accuracy as well as the worst-group test accuracy for each method and present the results in Table 3.5.

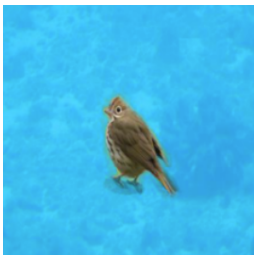
Our results demonstrate that Weighted Group DRO improves worst-group accuracy compared to both ERM and Group DRO models. Crucially, the success of our method hinges on accurately estimating group label probabilities,especially the minority groups.

In Figure 3.2, we present examples of images along with their predicted probabilities, demonstrating that higher group probability correlates with better class representation. These results highlight the effectiveness of Weighted Group DRO when group probabilities are accurately estimated. The key advantage of our method over standard Group DRO lies in its formulation, which assigns higher group weights to examples where $P(g|x_i, y_i)$ is high. Assuming that the model f_θ is accurate and well-calibrated, these examples are more likely to belong to the true group g. This approach allows us to focus on more representative examples of the true group, whereas standard Group DRO would assign equal weights. Our experimental results thus confirm the efficacy of Weighted Group DRO in scenarios with missing group labels.

		Avg accuracy			Worst Accuracy		
		ERM	GDRO	WGDRO	ERM	GDRO	WGDRO
CelebA	train	96.9%	93.4%	92.3%	46.2%	91.0%	91.3%
	test	95.7%	93.6%	92.5%	40.0%	80.6%	**87.8%**↑7%
Waterbirds	train	100%	100%	100%	100%	100%	100%
	test	84.5%	89.2%	87.1%	67.1%	75.5%	**77.4%**↑2%

Table 3.5: Comparison of ERM, Group DRO, and Weighted Group DRO across datasets with limited group labels. Green cells show our method's worst-group test accuracy. Arrows indicate improvement over Group DRO using binary versions of the same group assignments.

(a) Land bird on water background, prediction probability 0.53



(b) Land bird on water background, prediction probability 0.99

(c) Land bird on land background, prediction probability 0.99

(d) Land bird on land background, prediction probability 0.49

Figure 3.2: Images with true groups and predicted probabilities using $f_\theta = P(g|x, y)$. (a) and (b): Land birds on water; (b) shows higher probability due to visible water background, unlike (a)'s ambiguous setting. (c) and (d): Land birds on land; (c) is more representative, while (d)'s blue sky might be mistaken for water, resulting in higher predicted probability for the water group.

Conclusion

This chapter introduced methods for estimating group probabilities in scenarios with noisy or missing group labels. Our experiments revealed significant improvements in worst-group test accuracy under both noisy label conditions and limited group annotations, highlighting the advantages of probabilistic group assignments.

These findings open several avenues for future research: developing advanced techniques for group probability estimation, analyzing the relationship between predicted probabilities and downstream classification performance, and extending probabilistic group assignments to other subgroup robustness and out-of-domain generalization methods. The success of

our approach suggests that a probabilistic framework could enhance various techniques in robust machine learning, potentially leading to more reliable and generalizable models across diverse domains.

Chapter 4

Robust Decision Trees

Extensive research exists for domain generalization using linear models and neural networks, but few methods address domain generalization using decision trees. Tree-based models achieve competitive performance on many datasets, offering simplicity, computational efficiency, and often interpretability, making them popular for various applications. However, decision trees are susceptible to overfitting and tend to rely on features that may not generalize well to unseen data, especially as tree depth increases.

While methods such as random forests and gradient boosting trees improve in-distribution generalization, they do not adequately address out-of-distribution (OOD) generalization. A limited number of methods, such as those proposed by [11, 1], focus on robust decision trees. However, these approaches primarily address adversarial attacks rather than domain generalization. Tree-based methods, according to one study [21], inherently exhibit strong subgroup robustness. Nevertheless, these models often struggle to generalize effectively to out-of-distribution domains.

Our proposed method leverages the sequential nature of decision trees to learn invariant models. Since each split in the tree acts as a weak learner, by enforcing invariance at each level, we address a significant limitation of Invariant Risk Minimization (IRM). Specifically, this approach mitigates IRM's tendency to degenerate into Empirical Risk Minimization

(ERM) in overparameterized settings. We provide support for our proposed method through theoretical analysis and empirical evaluation.

4.1 Related Work

Domain Adaptation & Generalization: many approaches in this field aim to learn domain-invariant representations $Z = \phi(x)$ that maintain predictive power [7, 56, 43]. Techniques include using Maximum Mean Discrepancy loss [24, 39] and adversarial training [19, 23, 57] to minimize domain shift. The objective might enforce that $P(Z)$ as well as the predicted label distribution $P(Y|Z)$ to be the same across all the environments, which might not hold for some datasets [63, 2]. Subsequent methods including Conditional domain adversarial networks (CDAN) [40, 36, 55] seek to address this issue by conditioning on the label.

Distributional Robustness: this approach, formalized in equation (1.1), aims to minimize worst-case performance over a perturbation set, often defined using distance metrics like Wasserstein balls [51, 9]. Group DRO [50] extends this to minimize worst-group risk, where the groups are assumed to be mixtures of the training distributions. While effective for interpolation between training environments, these methods often struggle with extrapolation [2]. Risk Extrapolation (REx) [33] attempts to address this limitation by allowing affine, rather than just convex, combinations of training risks.

Invariant Models: recent OOD generalization research focuses on learning invariant models. Invariant Causal Prediction [46] approaches this from a causality perspective, distinguishing between stable causal features and spurious correlations. Invariant Risk Minimization (IRM) [3] enforces invariance in the learned representations as a method for identifying causal features. Rather than merely aligning the distributions of Z, IRM learns a representation $Z = \phi(X)$ such that the relationship $\mathbb{E}[y|\phi(x)]$ remains invariant across different environments. Subsequent research has proposed variations with stronger invariance constraints

[33, 61, 29, 42, 6]. However, these methods can face limitations, such as IRM reducing to Empirical Risk Minimization (ERM) when the model perfectly fits the training data [49].

Environment Inference: many of the approaches discussed above require access to samples from multiple environments, which can be challenging and costly to obtain. These environments should be designed to emphasize relevant variations in data distributions. Some research such as [14] addresses this by learning to split the data into environments that capture spurious correlations. This method employs a reference classifier and seeks to learn an environment partitioning that maximally violates the invariant learning principle. This approach draws inspiration from fairness techniques for handling unknown group memberships [31, 34], and is similar to methods used for identifying task groupings in multi-task learning [18]. Other strategies generate new environments by dynamically perturbing the data [29].

Out of Distribution Generalization with Trees: while ensemble methods like Random Forests and XGBoost [10, 30, 13] improve in-distribution generalization for trees, few methods address OOD generalization. Some research focuses on enhancing robustness against adversarial attacks [11, 1] by optimizing objectives that consider relevant examples within a predefined perturbation set, such as an l_∞ ball. Some work [12] also proves that formal robustness verification of decision trees can be achieved in linear time. Recent research [21] has also demonstrated strong subgroup robustness in tree-based methods, suggesting potential for out-of-distribution generalization.

4.2 Methodology

4.2.1 Problem Definition

We consider a supervised learning task, where we learn a mapping from the input features $X \in \mathcal{X}$ (where a single input data point is denoted $x \in \mathbb{R}^n$) to the labels $Y \in \mathcal{Y}$. The data originates from a set of E environments $\mathcal{E} = e_1, \ldots, e_E$, where $P_e(X^e, Y^e)$ denotes the data distribution corresponding to environment e. The set of environments $\mathcal{E}$ is partitioned into

seen training environments $\mathcal{E}_{\text{tr}}$ and unseen test environments $\mathcal{E}_{\text{te}}$. Our objective is to learn a model that generalizes well to the unseen test environments. We use the terms *environment*, *domain*, and *group* interchangeably throughout this work.

We assume that the input X^e is generated from a latent variable $Z^e = (Z^e_{\text{causal}}, Z^e_{\text{spur}})$, where in the cow classification example, the causal features might be the characteristics of the cow itself, and the spurious features could include the background. Following [3], we assume that the different environments are generated from the same structural equation model (SEM) but are formed with distinct interventions. Under this assumption, the relationship between the causal features and the label $P(Y|Z_{\text{causal}})$ remains invariant across environments, whereas $P(Y|Z_{\text{spur}})$ may vary.

4.2.2 Failures of Existing Invariant Methods

Invariant methods aim to learn features whose relationship with the label remains consistent across environments. Invariant Risk Minimization (IRM) learns a feature representation such that the optimal classifier on top of this representation is identical across environments. Consider a model comprising two components: a feature extractor ϕ and a predictor w operating on the extracted features. IRM seeks to enforce the following constraint: $w \in \arg\min_w \mathcal{L}^e(\phi, w), \quad \forall e \in \mathcal{E}_{\text{tr}}$. However, when the model is overparameterized and can perfectly fit the data, the invariance criterion becomes trivial to satisfy, and the model effectively reduces to Empirical Risk Minimization (ERM). Tree-based models can address this problem by enforcing stage-wise invariance. Each split in a decision tree essentially acts as a weak learner; thus, by enforcing invariance at each split, we can overcome this limitation.

4.2.3 Invariant Decision Tree Criteria

To enhance generalization, we focus on utilizing causal features that remain stable across different environments. To achieve this, Invariant Risk Minimization (4.1) [3] introduces a modified objective that learns a data representation $\phi(x)$ such that the optimal classifier

operating on it is consistent across all training environments. We define $\mathcal{L}^e(\phi, w)$ as the loss for data from environment e using feature representation ϕ and predictor w.

$$\begin{aligned} &\min_{\phi, w} \mathcal{L}(\phi, w) \\ &\text{s.t.} \quad w \in \arg\min_{\bar{w}} \mathcal{L}^e(\phi, \bar{w}) \quad \forall e \in \mathcal{E}_{\text{tr}} \end{aligned} \tag{4.1}$$

For invariant decision trees, we adopt a similar invariance criteria, which is enforced at every node of the tree. One splitting rule for internal nodes in a decision tree is based on thresholding the value of a single feature. When creating an invariant split, we select a (feature, threshold) combination such that the optimal threshold is consistent across environments according to criteria (4.2).

$$\begin{aligned} &\min_{f, t} \mathcal{L}(f, t) \\ &\text{s.t.} \quad t \in \arg\min_{\bar{t}} \mathcal{L}^e(f, \bar{t}) \quad \forall e \in \mathcal{E}_{\text{tr}} \end{aligned} \tag{4.2}$$

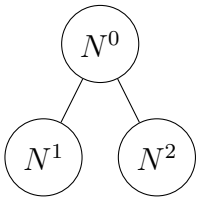
We define $\mathcal{L}^e(f, t)$ as the loss for data from environment e using the feature f and threshold t, and $\mathcal{L}(f, t)$ is the total loss using data from all training environments.

For a single cut, this invariance criteria parallels the IRM criteria, where selecting a feature f is analogous to learning the representation ϕ, and the threshold t corresponds to the classifier w. It is important to note that the formulation in (4.2) pertains to axis-parallel trees, as it requires selecting a single feature and its corresponding threshold. This concept can be extended to oblique decision trees [44], where instead of choosing a single feature f, we use a linear combination of multiple features, defined by a vector of coefficients.

4.2.4 Criteria Optimality

In this section, we justify the chosen tree invariance criteria and prove that any tree constructed according to this method will also satisfy the IRM invariance criteria. First, note

that for a single cut, tree invariance is equivalent to the invariance defined by IRM, so locally, a node in the tree that is chosen according to (4.2) satisfies invariance. To see this, recall that a single cut is determined by a (feature, threshold) pair, such that the optimal threshold is consistent across all training environments. This is analogous to IRM, where the chosen feature effectively serves as the node's internal data representation, and the threshold acts as the classifier operating on top of it. Therefore, a single decision stump that satisfies the tree invariance criteria (4.2) also satisfies the IRM criteria (4.1). Next, we demonstrate that a tree constructed as described above will satisfy the IRM invariance criteria (4.1). Consider the following data representation: for a tree with m nodes, the data representation is a vector of dimension $n \times m$, such that at node i, the representation $z^i = \phi^{(i)}(x)$ is given by $\left(z^i_{i \cdot n} = x_0, z^i_{i \cdot n+1} = x_1, \ldots, z^i_{i \cdot n+n} = x_n\right)$ and is zero everywhere else. For example, for a tree with three nodes, the representation is as follows:



$$\phi^{(0)}(x) = \begin{bmatrix} x_0 \\ \vdots \\ x_n \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \phi^{(1)}(x) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ x_0 \\ \vdots \\ x_n \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \phi^{(2)}(x) = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ x_0 \\ \vdots \\ x_n \end{bmatrix}$$

At node i, we operate on the representation $\phi^i(x)$ and optimize for a (feature, threshold) pair that satisfies (4.2). Note that at each node, we work with a subset of the data that has reached that node. Specifically, at node i for environment e, we define e^i as the subset of the environment data that reaches that node. We further define the optimal feature and threshold pair chosen to satisfy (4.2) as f^i, t^i, and this pair is optimized based on the representation ϕ^i.

Theorem 1. *An invariant tree with per-node representation ϕ^i, where cuts are chosen according to the invariant tree criteria (4.2), will also satisfy the IRM criteria (4.1).*

Proof. We show that if a split is locally invariant—i.e., the split at a specific node satisfies the invariance condition (4.2)—then it is also globally invariant, meaning it satisfies the IRM invariance criterion (4.1). Consider any single node N^i in the tree with its corresponding representation ϕ^i. Let (f^i, t^i) be a cut that operates on the representation ϕ^i and satisfies condition (4.2). This implies that the split is consistent across the environment data e^i associated with this node. To demonstrate that this results in IRM optimality across the original environments, denote the environment data that reached node N^i as e^i and the data that did not as e^{-i}. The union of these gives the original environment $e = e^i \cup e^{-i}$. Since all data from environments e^{-i} was encoded using ϕ^{-i}, which differs from ϕ^i, it must be zero wherever ϕ^i is non-zero. Consequently, the value corresponding to feature f^i is zero for all data from e^{-i}, rendering it trivially invariant (as any representation that is zero everywhere is invariant). For the data from e^i, the cut is invariant by construction. Therefore, this cut must be invariant with respect to the entire set of training environments, satisfying condition (4.1). □

4.3 Failure Modes

4.3.1 Number of Examples Per Environment

Similar to many other domain generalization methods, an invariant tree's performance is sensitive to the number of examples available per environment. Specifically, if the environments contain very few data points, the method is likely to fail. For instance, consider an extreme scenario where each environment contains only a single e data point. In this case, since the method aims to learn a representation invariant across environments, it essentially searches for a representation that is invariant across individual data points, as they define the environments. Furthermore, because tree-based methods partition the dataset so that

each node operates on a subset of the data, some nodes—particularly those near the bottom of the tree—may have few or no data points for certain environments. In such cases, the invariance criterion can be trivially satisfied, reducing the method to empirical risk minimization (ERM) at that node. This issue arises not only at the level of the initial dataset but also within each node of the tree, as each node operates on a progressively smaller subset of the data.

4.3.2 Invariance Cannot Be Achieved

The proposed method aims to find a feature-threshold pair such that the same threshold is optimal across all environments. However, this definition of invariance is strict, and the method may struggle to find any pair that satisfies this criteria. To address this issue, one approach is to modify the formulation using Lagrangian relaxation. Alternatively, introducing more flexibility in the internal node representation can mitigate this problem. Instead of selecting a single feature, we can allow for oblique cuts (a linear combination of multiple features) or use a non-linear approach. This added flexibility can make it easier to identify an invariant representation.

4.4 Regret Minimization Tree

In this section, we formulate a variation of the invariant decision tree method, which we call the regret minimization tree.

Invariant Risk Minimization (IRM) seeks a representation such that the optimal classifier built upon it remains invariant to domain changes. In the previous sections, we defined a cut (feature, threshold) pair in a tree as invariant if the optimal threshold is the same across all environments. However, it is likely that for some nodes, a strictly invariant cut does not exist. To address this, we can use Lagrangian relaxation of this criterion, selecting a cut that balances the trade-off between training performance and invariance. It can be formulated

as:

$$\min_{f,t} \mathcal{L}(f,t) + \lambda \sum_{e \in \mathcal{E}_{\text{tr}}} \left(\mathcal{L}^e(f,t) - \min_{\bar{t}} \mathcal{L}^e(f,\bar{t}) \right) \tag{4.3}$$

Where $\mathcal{L}(f,t)$ is the loss using feature f, threshold t, evaluated on data from all environments, and $\mathcal{L}^e(f,t)$ is the loss evaluated on data from environment e only.

If for a given feature f and environment e we refer to t as the optimal threshold trained on all the data *including* samples from environment e, and t^e as the optimal threshold for environment e, then the regularizing term can be written as $(\mathcal{L}^e(f,t) - \mathcal{L}^e(f,t^e))$ effectively measuring how sub-optimal threshold t is for environment e.

The IRM principle has inspired methods such as regret minimization [29], which replaces the invariance criterion with a regret minimization objective, addressing some of the limitations of IRM. In our method, we define the regret for a given environment and feature as the difference between the loss using the threshold t^{-e} trained on all environments *except* e, and the loss using t^e, the optimal threshold for environment e:

$$\mathcal{R}^e(f) = \mathcal{L}^e(f,t^{-e}) - \mathcal{L}^e(f,t^e) \tag{4.4}$$

where the thresholds are determined by

$$t^{-e} = \arg\min_{\bar{t}} \sum_{k \in \mathcal{E}_{\text{tr}} \setminus \{e\}} \mathcal{L}^k(f,\bar{t}) \qquad t^e = \arg\min_{\bar{t}} \mathcal{L}^e(f,\bar{t}) \tag{4.5}$$

It is important to note that the predictor t^e is both trained and evaluated on environment e, whereas the predictor t^{-e} is trained on on all environments except for environment e and evaluated on environment e. The regret is non-negative by definition, i.e., regret ≥ 0, and minimizing this gap aims to reduce the performance disparity between these predictors. Even if we achieve zero training loss for t^{-e} across all environments except e, i.e., $\mathcal{L}^k(f,t^{-e}) = 0, \forall k \in \mathcal{E} \setminus e$, it may still generalize poorly to environment e (i.e., $\mathcal{L}^e(f,t^{-e}) > 0$).

Consequently, the zero regret criterion imposes a more stringent requirement than the Invariant Risk Minimization (IRM) constraint, which should improve generalization in the overparameterized regimen.

Ideally, we aim to achieve zero regret across all environments. Since such a cut might not exist, we employ Lagrangian relaxation and define the following optimization problem, where λ balances accuracy and invariance: $\lambda = 0$ corresponds to ERM, while $\lambda \to \infty$ minimizes the regret.

$$f^*, t^* = \arg\min_{f,t} \mathcal{L}(f, t) + \lambda \sum_{e \in \mathcal{E}_{tr}} \mathcal{R}^e(f) \tag{4.6}$$

Algorithm 3 Regret Guided Node Split

Input: Data $\mathcal{D}_i = \{(x_j, y_j, e_j)\}_{j=1}^{N_i}$ arriving at node i
Input: Feature set F_i and training environment set $\mathcal{E}_i$ for node i
Input: Regularization parameter λ
Define: $\mathcal{D}_i^e = \{(x_j, y_j, e_j) \in \mathcal{D}_i : e_j = e\}$: data from environment e at node i
Define: $\mathcal{D}_i^{\mathcal{E}_i \setminus \{e\}} = \{(x_j, y_j, e_j) \in \mathcal{D}_i : e_j \neq e\}$: data from all environments except e at node i
Define: $\mathcal{L}(f, t, \mathcal{D})$ loss using data D, feature f, and threshold t

```
for each feature f ∈ F_i do
    for each environment e ∈ E_i do
        t^{-e} = arg min_t L(f, t, D_i^{E_i \ {e}})   ▷ Optimal threshold excluding environment e
        t^e = arg min_t L(f, t, D_i^e)                ▷ Optimal threshold for data from environment e
        R^e(f) = L(f, t^{-e}, D_i^e) − L(f, t^e, D_i^e)   ▷ Regret for environment e data
    end for
end for
f, t ← arg min_{f,t} L(f, t, D_i) + λ Σ_{e∈E_i} R^e(f)
Split node N using feature f and threshold t
```

Optimization of a Decision Stump

To calculate the regret for each environment e, we determine the optimal threshold for the environment: t^e, and the threshold calculated using data from all other environments excluding e : t^{-e}. We then evaluate the loss for data from environment e using these thresholds, and compute the environment-specific regret. Additionally, we calculate the total Gini impu-

Env	1	2	3	4	5	6	7	8	9	10	11	12	13		d
0.1	.28	.21	.21	.23	.23	.18	.17	.14	.13	.09	.07	.06	.01	$\cdots$	0.0
0.2	.36	.28	.31	.35	.35	.31	.27	.2	.18	.13	.1	.12	.09	$\cdots$	0.0
0.9	.34	.19	.14	.13	.14	.13	.12	.1	.08	.06	.06	.06	.05	$\cdots$	0.0

Table 4.1: Example of regret measurements of a standard decision tree, using ColoredMNIST dataset. The vertical axis is the environment specification, and the horizontal axis is the depth of the tree. We present the average regret measurement across all nodes in this depth. The regret is generally decreasing as the depth increases.

rity using data from all environments combined. It is important to note that Gini impurity is not additive, meaning we cannot simply calculate the Gini impurity for each individual environment and take their weighted average. This is because it is possible to find a split that results in zero Gini impurity for each environment individually, but the combined Gini impurity remains positive.

4.5 How Invariant Are Decision Trees?

We aim to analyze the extent to which a standard decision tree classifier is invariant. To achieve this, we train a standard decision tree (without invariance constraints) on the complete dataset. We then examine the resulting tree and evaluate the invariance of each cut, using the regret as a proxy. At each node, for each environment, we measure the regret—defined as the performance gap between the optimal split for the given environment and the split chosen by the classifier.

In Table 4.1 we present an example of the regret (measured by the Gini gain difference; for a complete description of the regret calculation, see 4.4) across different environments and tree depths in the Colored MNIST dataset for analysis purposes. Note the lower regret closer to the leaves of the tree, where we have fewer examples per environment.

4.5.1 Tree Pruning

The previous section proposed a top-down approach for constructing an invariant tree. An alternative method is to first construct the full decision tree (or a few levels of it) and then use pruning to remove nodes that violate the invariance criteria. This approach starts at the leaves of the tree, checking whether they are invariant—i.e., whether they satisfy the criteria (4.2). If a leaf does not satisfy the invariance criteria, it is pruned, and the process continues by enforcing invariance at higher levels of the tree. Conversely, if the cuts satisfy the invariance criteria, they are retained in the tree.

The advantages of this method include improved efficiency, as we only need to check whether an existing node is invariant (or close to invariant) rather than iterating over all possible combinations to check if an invariant cut exists. Additionally, this approach can be applied to any existing tree, enhancing its generalization, making it flexible and easy to adapt. However, a major issue with this method is that it starts from the bottom of the tree, where nodes operate on small subsets of the data. In these cases, it can be trivial to satisfy the invariance criteria. For example, if a leaf node contains only a few examples, they might all come from a single environment, making invariance meaningless, and thus the tree remains unchanged after pruning. Because it is easier to satisfy the invariance criteria near the leaves of the tree compared to the root, starting from the bottom may prevent the pruning of non-invariant nodes that are closer to the root.

While this method may not always be optimal, it offers significant flexibility and ease of use. By pruning nodes at the bottom of the tree, we reduce model complexity. Additionally, pruning can be applied during the construction of the tree, after building only a few levels, which partially addresses the issues mentioned earlier. From experiments to determine whether an invariant substructure exists (4.1), it is evident that some nodes are invariant while others are not, so pruning can be feasible and useful.

4.6 Experiments

For the experiments outlined below, we use the regret minimization method described in Section 4.4. For the standard decision tree and random forest classifiers, we use `Sklearn DecisionTreeClassifier` and `Sklearn RandomForestClassifier` [45]. For our regret minimization tree, we implement a standard decision tree algorithm and use algorithm 3 to optimize the feature and threshold at each node. Additional data and implementation details are provided in Appendix A.

4.6.1 Synthetic Dataset

We begin by validating our hypothesis using a synthetic dataset, generated following a procedure proposed by [3]:

$$
\begin{aligned}
X_c &\leftarrow \text{Gaussian}(0, e^2) \\
Y &\leftarrow X_c + \text{Gaussian}(0, e^2) \\
X_{nc} &\leftarrow Y + \text{Gaussian}(0, 1)
\end{aligned}
$$

This Structural Equation Model (SEM) creates data where the correlation between X_{nc} and Y is stronger than the correlation between X_c and Y. Consequently, a model trained with an Empirical Risk Minimization (ERM) objective is likely to rely on the non-causal features, while an invariance constraint encourages the model to focus on the causal features.

Following [3], we use the SEM to generate data. Each experiment draws samples from three training environments $\mathcal{E}_{\text{tr}} = 0.2, 2, 5$. Following the original paper, we generate 1000 data samples for each environment. We follow the same procedure and use the environment $e = 5$ for evaluation. We compare the performance of a standard decision tree classifier, a random forest classifier, and a single decision tree trained with a regret minimization objective and lambda value of $1,000$, which is a hyperparameter. The maximum depth of

the standard decision tree and the random forest are set to 10. The maximum depth of the regret minimization tree is set to 4. We use 10 estimators in the Random Forest experiments. For hyperparameters tuning we use the training-domain validation set (as suggested in [25]), selecting 10% of each training environment for validation. The final results were measured on a held-out environment.

The Regret Minimization Tree achieves the best performance, with a test accuracy of **0.88**, compared to **0.36** for the single decision tree and **0.83** for the random forest model. Furthermore, the invariant tree exhibits the smallest generalization gap between training and testing, which is advantageous as the training performance more accurately predicts the test performance.

4.6.2 Heart Failure Dataset

We use the heart failure dataset [15] which consists of four independently collected datasets, where each dataset is considered a separate environment. Each environment is taken from a different hospital which follows different protocols, so the distributions $P(Y)$, $P(X)$, and $P(Y|X)$ can vary between environments, but it is reasonable to assume that there is some underlying consistent classification rule $P(Y|X_C)$ that is stable across environments. The task is a binary heart failure prediction task based on 13 available features. Missing values are set to -1. The environment counts and results are presented in Table 4.2.

We perform four separate experiments, each time selecting a different environment for testing. We select hyperparameters using leave-one-domain-out cross-validation, as proposed in [25]. We use max tree depth of 10 for the standard tree and random forest and tree depth of 5 for the regret minimization tree. Random forest uses 10 estimators. For regret minimization, λ is set to a high value of 1000, which was also selected using the validation data.

Test Env	Observations	Decision Tree	Random Forest	Regret Tree
Cleveland	303	0.96, 0.67	0.96, **0.72**	0.92, 0.67
Hungarian	294	0.96, 0.72	0.98, 0.71	0.8, **0.76**
Switzerland	123	0.95, 0.54	0.96, 0.72	0.8, **0.85**
Long Beach	200	0.98, 0.59	0.97, 0.68	0.84, **0.7**
Average	–	0.96, 0.63	0.97, 0.71	0.84, **0.75**

Table 4.2: Experimental results for heart failure data: (train, test) accuracy. The highlighted column (Regret Tree) represents our proposed method.

4.6.3 Conclusion

The results are promising, demonstrating test performance improvements for regret minimization trees compared to both standard decision trees and random forests. This indicates that regret minimizing trees are successfully enhancing out-of-domain generalization. However, to strengthen these findings, further experiments should be conducted. These should include evaluating performance on a wider range of datasets and assessing the performance of an ensemble of invariant trees.

Chapter 5

Conclusion

In this work, we introduce two novel approaches to enhance model robustness. First, we propose Weighted Group DRO (WGDRO), an alternative formulation to Group DRO that improves performance when group assignment is probabilistic. We present two methods for estimating group probabilities and demonstrate their empirical effectiveness. Weighted Group DRO is applicable in scenarios where Group DRO is traditionally used, such as improving subgroup robustness and domain generalization. Our empirical results support the method's effectiveness in enhancing subgroup robustness under two challenging conditions: noisy group labels and limited group annotations.

Second, we address decision tree robustness by proposing an invariant decision tree formulation and developing more practical variations of this method. We provide theoretical proofs for the optimality of our criteria and support our claims with empirical evidence.

These approaches contribute to the field of robust machine learning, offering potential tools and insights for researchers and developers dealing with uncertain group assignments and aiming to enhance decision tree performance across domains.

Appendix A

Implementation Details

A.0.1 Data

We use CelebA and Waterbirds datasets with the group assignments as created in a previous Group DRO paper [50], dividing each dataset into four groups according to the label, attribute combination. For CelebA we use $\{gender, haircolor\}$ label attribute combination do define groups, and for Waterbirds we use $\{birdtype, backgroundtype\}$.

We then modify the group into a noisy group label by adding noise using the following methods.

Uniform Noise:

- With probability g, we keep the group label unchanged $\tilde{g} = g$.
- With probability $1 - g$, we sample a new group label uniformly at random from the remaining $G - 1$ groups.

We repeat this process with noise levels $p = 0.1$ and $p = 0.3$.

Variable noise:

- With probability g, we keep the group label unchanged $\tilde{g} = g$.
- With probability $1 - g$, we sample a new group label using the following procedure, that depends on the value of the true group label:

1. We define 50% of the groups to be in the set G_{uniform}, and if the true group is in G_{uniform}, we proceed to sampling the noisy group label uniformly at random from the remaining $G-1$ groups.
2. We define 50% of the groups to be in the set G_{biased}, and if the true group is in G_{biased}, we proceed to sampling the noisy group label such that its noisy group assignment twice as likely to belong to one of the groups compared to the remaining $G-2$ groups.

We use $p = 0.1$.

A.0.2 Weighted Group DRO Implementation

Weighted GDRO Model:

We train a ResNet50 model using pytorch `torchvision` implementation with pre-trained weights. The model takes $\{x_i, y_i, q_i(g)\}_{i=1}^n$ as input and optimizes the Weighted Group DRO objective using $q_i(g)$. We train the model using stochastic gradient descent with a momentum term of 0.9 and a batch size of 128. As in the previous Group DRO paper, we used batch normalization [28] and no dropout [54]. For simplicity, we train all models without data augmentation. We use learning rate 0.0001 for celebA and 0.001 for Waterbirds with weight decay 0.0001, keeping the same hyperparameters as uused in the previous paper. We train for 10 epochs for CelebA and 20 epochs for Waterbirds and select the model with the highest worst-group validation performance.

Estimating Group Probabilities

Group estimation with noisy groups

For estimating $P(g|x_i, y_i, \tilde{g}_i)$ we following different methods:

- **Group Label Smoothing (WGDRO-S):** we simplify $P(g|x_i, y_i, \tilde{g}_i)$ to $P(g|\tilde{g}_i)$, assuming instance independent noise model. We estimate it using a simple label smooth-

ing approach, as described in 3.1.3. We use $\alpha = 0.05$, where α is a hyper-parameter that was chosen using the validation data performance.

- **Parameterized Group Estimation using EM Algorithm (WGDRO-EM):** this appendix provides a detailed description of the implementation of the Expectation-Maximization (EM) algorithm for learning true group labels from noisy observations.

  - **Model Architecture:** we use `pytorch`, the group classifier is implemented as a Convolutional Neural Network (CNN) with the following structure: Three convolutional layers with ReLU activation and max pooling, followed by adaptive average pooling, followed by two sets of fully connected layers - one for processing the image features, and another for combining image features with the label y. The exact architecture is as follows:

    * First Convolutional Block:

      2D Convolution: 32 filters, 3x3 kernel, padding=1

      ReLU activation

      Max Pooling: 2x2

    * Second Convolutional Block:

      2D Convolution: 64 filters, 3x3 kernel, padding=1

      ReLU activation

      Max Pooling: 2x2

    * Third Convolutional Block:

      2D Convolution: 64 filters, 3x3 kernel, padding=1

      ReLU activation

      Max Pooling: 2x2

    * Output Layer:

      Adaptive Average Pooling: output size 4x4

    * Image Fully Connected Layers:

Input 1024 (64 * 4 * 4), Output 128 with ReLU activation.

* Combined Fully Connected Layers:

  FC layer with input size $129(128+1)$, Output 64. Note: The additional 1 in the input corresponds to a single y value. ReLU activation.

  Fully connected layer: Input 64, Output G

– **Objective:** as described in the main text and fully derived in Appendix B, we use the objective in Algorithm 2.

– **Hyperparameters:** We use Adam optimizer, with learning rate 0.001 for both datasets and weight decay 0.0001. The learning rate was selected after experimenting with different learning rates ($1e^{-2}$, $1e^{-3}$, $1e^{-4}$) and monitoring the loss.

– **Initialization:** the noise model is set to

$$P(\tilde{g}|g) = \begin{cases} 0.9 & \text{if } g = \tilde{g} \\ \frac{0.1}{G-1} & \text{otherwise} \end{cases}$$

The model parameters are initialized with the PyTorch default weights.

– **Iterations:** We trained the model for 20 epochs for CelebA and 40 epochs for waterbirds which was set by monitoring the loss and the likelihood, and used the last model in the run.

– **Group Prediction Evaluation:** During the experiments we evaluate the quality of the group assignments using the downstream classification model worst-group validation accuracy as a proxy.

Algorithm performance: after finalizing the experiments we evaluate the group assignment prediction on the training set by calculating precision and recall. The results are shown in Appendix C.

Group estimation from small labeled dataset

In this settings we use a small dataset labeled with the true group assignments $D_l = \{(x_i, y_i, g_i)\}_{i=1}^{n_l}$ to learn a model $P(g|x_i, y_i; \theta)$. We have access to 10% of the training data along with its true group annotation, where the examples are sampled in a stratified way, making the sampling probability of the 2 minority groups twice their percentage in the dataset, increasing the number of minority examples. We use a CNN architecture with the following structure:

Feature Extractor:

- First Convolutional Block:

  2D Convolution: 32 filters, 3×3 kernel, padding=1

  ReLU activation

  Max Pooling: 2×2

- Second Convolutional Block:

  2D Convolution: 64 filters, 3×3 kernel, padding=1

  ReLU activation

Classifier:

- Fully Connected Layer: 65 (64 from CNN + 1 additional input), ReLU activation, output size 128
- Output Layer: size 128 to size G

We use Adam optimizer with learning rate 0.001 (tested $1e^{-2}$, $1e^{-3}$, $1e^{-4}$ and monitored the loss) and batch size 128. We run the model for 100 epochs, and use train accuracy as well as per group precision and recall to evaluate the performance. We then use the learned model to estimate the group probabilities for all the training dataset, and use it as the group weights $q_i(g)$, and use the validation performance for evaluation.

A.0.3 Invariant Decision Trees

Additional Dataset Information

The heart failure dataset is a prediction task, where the target values range from 0 (no presence) to 4, and is binarized to distinguish presence (values 1,2,3,4) from absence (value 0). Binary labels have the following target distribution as seen in Table A.1.

Database	**class 0**	**class 1**
Cleveland	164	139
Hungarian	188	106
Switzerland	8	115
Long Beach VA	51	149

Table A.1: Target class distribution for each heart failure environment

A.0.4 Implementation

We use sklearn DecisionTreeClassifier for the standard decision tree, using the default parameters. For random forest we use sklearn RandomForestClassifier with the default parameters. We implement our invariant decision tree from scratch using python, with the Gini Impurity score as the regret measure, and max tree depth of 4 and 5 for the synthetic and heart-failure datasets respectively. We set $\lambda = 1000$. We choose these hyperparameters in a standard manner, as described in the main text. We evaluate the performance of the final model on a held out test environments.

Appendix B

Derivations

B.1 Group probability calculation derivation

Derivation for $P(g|x_i, y_i, \tilde{g}_i) = P(\tilde{g}|x, y, g)\frac{P(g|x,y)}{P(\tilde{g}|x,y)}$:

$$\begin{aligned} P(g|x_i, y_i, \tilde{g}_i) &= \frac{P(x, y, \tilde{g}, g)}{P(x, y, \tilde{g})} \\ &= P(\tilde{g}|x, y, g)\frac{P(x, y, g)}{P(x, y, \tilde{g})} \\ &= P(\tilde{g}|x, y, g)\frac{P(g|x, y)P(x, y)}{P(x, y, \tilde{g})} \\ &= P(\tilde{g}|x, y, g)\frac{P(g|x, y)}{P(\tilde{g}|x, y)} \end{aligned}$$

B.2 EM objective derivation

EM Objective: our goal is to estimate distribution of the latent variable g, i.e. $P(g|x, y, \tilde{g})$. Given the relationship $P(g|x, y, \tilde{g}) = P(\tilde{g}|x, y, g)\frac{P(g|x,y)}{P(\tilde{g}|x,y)} \propto P(\tilde{g}|x, y, g)P(g|x, y)$, in order to calculate $P(g|x, y, \tilde{g})$, we need to estimate $P(g|x, y)$ and have the noise model $P(\tilde{g}|x, y, g) = P(\tilde{g}|g)$ assuming that we have instance-independent noise.

We use the EM algorithm to estimate $P(g|x_i, y_i; \theta)$ in a parametric way. We learn $P(g|x_i, y_i; \theta)$ using EM approach. As explained in the document, our goal is to maximize the log likelihood of the data:

$$L(\theta) = \sum_{i=1}^{n} \log p(\tilde{g}_i|x_i, y_i; \theta) = \sum_{i=1}^{n} \log(\sum_{g \in G} p(\tilde{g}_i|g) p(g|x_i, y_i; \theta))$$

However, it is not possible to directly set the derivative of this formula with respect to the parameters θ to zero and solve it in closed form. EM allows us to construct a lower bound on the log likelihood $L(\theta)$ (E-step) and then optimize the lower bound (M-step).

Further note that we proceed by considering the optimization of a single example $\log p(\tilde{g}|x, y; \theta) = \log \sum_{g \in G} p(\tilde{g}|x, y; \theta)$, and later on we will bring back the summation over i.

$$\log p(\tilde{g}|x, y; \theta) = \log \left[\sum_{g \in G} p(g|x, y; \theta) p(\tilde{g}|g) \right] \tag{B.1}$$

$$= \log \left[\sum_{g \in G} Q(g|x, y, \tilde{g}) \frac{p(g|x, y; \theta) p(\tilde{g}|g)}{Q(g|x, y, \tilde{g})} \right] \tag{B.2}$$

$$\geq \sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[\frac{p(g|x, y; \theta) p(\tilde{g}|g)}{Q(g|x, y, \tilde{g})} \right] \tag{B.3}$$

We then rewrite this equation as

$$\sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[p(g|x, y; \theta) p(\tilde{g}|g) \right] + \sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[\frac{1}{Q(g|x, y, \tilde{g})} \right]$$

And since the second term is independent of θ we can drop it and get the objective:

$$\sum_{g \in G} Q(g|x, y, \tilde{g}) \log \left[p(g|x, y; \theta) p(\tilde{g}|g) \right]$$

Dropping $p(\tilde{g}|g)$ since it doesn't depend on θ, we optimize the following:

$$\sum_{g \in G} Q(g|x, y, \tilde{g}) \log p(g|x, y; \theta)$$

We define $Q(g|x, y, \tilde{g}; \theta_0)$ as our previous best estimate of the group probabilities, and $p(g|x, y; \theta)$ is the model prediction using the most recent parameters θ. So from this we derive the EM algorithm steps:

- **E-step:** calculate the true group labels based on the current parameter values

$$q_i(g) = p(g|x_i, y_i, \tilde{g}_i; \theta)$$

- **M-step**: update the parameters θ using

$$\sum_{i=1}^{n} \sum_{g \in G} q_i(g) \log p(g|x_i, y_i; \theta)$$

We optimize the M-step using Cross Entropy loss between $q_i(g)$ and the model predictions.

Appendix C

Additional Experiments and Results

C.1 Data Statistics

Dataset	Sample Count (n)			
	waterbird = 0 background = 0	waterbird = 0 background = 1	waterbird = 1 background = 0	waterbird = 1 background = 1
Training	3498	184	56	1057
Validation	467	466	133	133
Test	2255	2255	642	642

Table C.1: Distribution of samples across training, validation, and test datasets in Waterbirds

Dataset	Sample Count (n)			
	Blond Hair = 0 Male = 0	Blond Hair = 0 Male = 1	Blond Hair = 1 Male = 0	Blond Hair = 1 Male = 1
Training	71629	66874	22880	1387
Validation	8535	8276	2874	182
Test	9767	7535	2480	180

Table C.2: Distribution of samples across training, validation, and test datasets in CelebA

C.1.1 EM for estimating groups from noisy groups

The EM algorithm achieves the following performance for predicting group label:

Uniform noise with probability 0.1:

Dataset	Metric	Accuracy	Precision	Recall
CelebA	Overall	97.96%	-	-
	Group 0	96.93%	98.67%	96.93%
	Group 1	98.60%	96.77%	98.60%
	Group 2	99.60%	99.58%	99.60%
	Group 3	92.86%	93.47%	92.86%
Waterbird	Overall	97.83%	-	-
	Group 0	99.23%	99.11%	99.23%
	Group 1	83.70%	85.56%	83.70%
	Group 2	37.50%	84.00%	37.50%
	Group 3	98.86%	96.05%	98.86%

Table C.3: Model performance for CelebA and Waterbird datasets using uniform noise with probability 0.1

Variable noise with probability 0.1:

Dataset	Metric	Accuracy	Precision	Recall
CelebA	Overall	98.22%	-	-
	Group 0	97.45%	98.71%	97.45%
	Group 1	98.64%	97.31%	98.64%
	Group 2	99.74%	99.57%	99.74%
	Group 3	92.93%	95.62%	92.93%
Waterbird	Overall	97.83%	-	-
	Group 0	99.23%	99.11%	99.23%
	Group 1	83.70%	85.56%	83.70%
	Group 2	37.50%	84.00%	37.50%
	Group 3	98.86%	96.05%	98.86%

Table C.4: Model performance for CelebA and Waterbird datasets, using Variable noise with probability 0.1

Bibliography

[1] Maksym Andriushchenko and Matthias Hein. Provably robust boosted decision stumps and trees against adversarial attacks. *arXiv preprint arXiv:1906.03526*, 2019.

[2] M Arjovsky. Out of distribution generalization in machine learning 2021. *arXiv preprint arXiv:2103.02667*, 2021.

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

[4] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018.

[5] Alan Joseph Bekker and Jacob Goldberger. Training deep neural-networks based on unreliable labels. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2682–2686. IEEE, 2016.

[6] Alexis Bellot and Mihaela van der Schaar. Accounting for Unobserved Confounding in Domain Generalization. *arXiv e-prints*, page arXiv:2007.10653, July 2020.

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

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

[9] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton University Press, 2009.

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

[11] Hongge Chen, Huan Zhang, Duane Boning, and Cho-Jui Hsieh. Robust decision trees against adversarial examples. In *International Conference on Machine Learning*, pages 1122–1131. PMLR, 2019.

[12] Hongge Chen, Huan Zhang, Si Si, Yang Li, Duane Boning, and Cho-Jui Hsieh. Robustness verification of tree-based models, 2019.

[13] Tianqi Chen and Carlos Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 785–794, New York, NY, USA, 2016. ACM.

[14] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR, 2021.

[15] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[16] John C Duchi, Peter W Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 46(3):946–969, 2021.

[17] Logan Engstrom, Brandon Tran, Dimitris Tsipras, Ludwig Schmidt, and Aleksander Madry. Exploring the landscape of spatial robustness. In *International Conference on Machine Learning*, pages 1802–1811. PMLR, 2019.

[18] Christopher Fifty, Ehsan Amid, Zhe Zhao, Tianhe Yu, Rohan Anil, and Chelsea Finn. Efficiently identifying task groupings for multi-task learning. *arXiv preprint arXiv:2109.04617*, 2021.

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

[20] 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.

[21] Josh Gardner, Zoran Popovic, and Ludwig Schmidt. Subgroup robustness grows on trees: An empirical baseline investigation. *Advances in Neural Information Processing Systems*, 35:9939–9954, 2022.

[22] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[23] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European conference on computer vision*, pages 597–613. Springer, 2016.

[24] A. Gretton, AJ. Smola, J. Huang, M. Schmittfull, KM. Borgwardt, and B. Schölkopf. *Covariate shift and local learning by distribution matching*, pages 131–160. MIT Press, Cambridge, MA, USA, 2009.

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

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

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[29] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Enforcing predictive invariance across structured biomedical domains. *arXiv preprint arXiv:2006.03908*, 2020.

[30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[31] Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.

[32] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

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

[34] Preethi Lahoti, Alex Beutel, Jilin Chen, Kang Lee, Flavien Prost, Nithum Thain, Xuezhi Wang, and Ed H Chi. Fairness without demographics through adversarially reweighted learning. *arXiv preprint arXiv:2006.13114*, 2020.

[35] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

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

[37] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.

[38] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15(2018):11, 2018.

[39] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015.

[40] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *arXiv preprint arXiv:1705.10667*, 2017.

[41] Michal Lukasik, Srinadh Bhojanapalli, Aditya Menon, and Sanjiv Kumar. Does label smoothing mitigate label noise? In *International Conference on Machine Learning*, pages 6448–6458. PMLR, 2020.

[42] Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, pages 7313–7324. PMLR, 2021.

[43] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pages 10–18. PMLR, 2013.

[44] Sreerama K Murthy, Simon Kasif, and Steven Salzberg. A system for induction of oblique decision trees. *Journal of artificial intelligence research*, 2:1–32, 1994.

[45] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

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

[47] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International Conference on Machine Learning*, pages 5389–5400. PMLR, 2019.

[48] Mamshad Nayeem Rizve, Kevin Duarte, Yogesh S Rawat, and Mubarak Shah. In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. *arXiv preprint arXiv:2101.06329*, 2021.

[49] Elan Rosenfeld, Pradeep Ravikumar, and Andrej Risteski. The risks of invariant risk minimization. *arXiv preprint arXiv:2010.05761*, 2020.

[50] 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.

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

[52] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International conference on machine learning*, pages 5907–5915. PMLR, 2019.

[53] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE transactions on neural networks and learning systems*, 34(11):8135–8153, 2022.

[54] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[55] Remi Tachet des Combes, Han Zhao, Yu-Xiang Wang, and Geoffrey J Gordon. Domain adaptation with conditional distribution matching and generalized label shift. *Advances in Neural Information Processing Systems*, 33, 2020.

[56] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE international conference on computer vision*, pages 4068–4076, 2015.

[57] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7167–7176, 2017.

[58] Abraham Wald. Statistical decision functions which minimize the maximum risk. *Annals of Mathematics*, 46(2):265–280, 1945.

[59] Serena Wang, Wenshuo Guo, Harikrishna Narasimhan, Andrew Cotter, Maya Gupta, and Michael Jordan. Robust optimization for fairness with noisy protected groups. *Advances in neural information processing systems*, 33:5190–5203, 2020.

[60] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699, 2015.

[61] Chuanlong Xie, Fei Chen, Yue Liu, and Zhenguo Li. Risk variance penalization: From distributional robustness to causality. *arXiv e-prints*, pages arXiv–2006, 2020.

[62] YiFan Zhang, Xue Wang, Jian Liang, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. Free lunch for domain adversarial training: Environment label smoothing. *arXiv preprint arXiv:2302.00194*, 2023.

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