No Subclass Left Behind: Fine-Grained Robustness in Coarse-Grained Classification Problems

Nimit S. Sohoni, Jared A. Dunnmon, Geoffrey Angus, Albert Gu, Christopher Ré

Stanford University
nims@stanford.edu, jdunnmon@cs.stanford.edu, gdlangus@cs.stanford.edu,
albertgu@stanford.edu, chrismre@cs.stanford.edu

December 1, 2020

Abstract

In real-world classification tasks, each class often comprises multiple finer-grained "subclasses." As the subclass labels are frequently unavailable, models trained using only the coarser-grained class labels often exhibit highly variable performance across different subclasses. This phenomenon, known as *hidden stratification*, has important consequences for models deployed in safety-critical applications such as medicine. We propose George, a method to both measure and mitigate hidden stratification *even when subclass labels are unknown*. We first observe that unlabeled subclasses are often separable in the feature space of deep models, and exploit this fact to estimate subclass labels for the training data via clustering techniques. We then use these approximate subclass labels as a form of noisy supervision in a distributionally robust optimization objective. We theoretically characterize the performance of George in terms of the worst-case generalization error across any subclass. We empirically validate George on a mix of real-world and benchmark image classification datasets, and show that our approach boosts worst-case subclass accuracy by up to 22 percentage points compared to standard training techniques, without requiring any information about the subclasses.

1 Introduction

In many real-world classification tasks, each labeled class consists of multiple semantically distinct subclasses that are unlabeled. Because models are typically trained to maximize *global* metrics such as average performance, they often underperform on important subclasses [52, 40]. This phenomenon—recently termed *hidden stratification*—can lead to skewed assessments of model quality and result in unexpectedly poor performance when models are deployed [36]. For instance, a medical imaging model trained to classify between benign and abnormal lesions may achieve high overall performance, yet consistently mislabel a rare but critical abnormal subclass as "benign" [17].

Modern robust optimization techniques can improve performance on poorly-performing groups when the group identities are known [43]. However, in practice, a key obstacle is that *subclasses are often unlabeled*, or even unidentified. This makes even detecting such performance gaps—let alone mitigating them—a challenging problem. Nevertheless, recent empirical evidence [36] encouragingly suggests that feature representations of deep neural networks often carry information about unlabeled subclasses (see Figure 1). Motivated by this observation, we propose a method for addressing hidden stratification, by both measuring and improving worst-case subclass performance in the setting where subclass labels are unavailable. Our work towards this is organized into four main sections.

First, in Section 3 we propose a simple generative model of the data labeling process. Using this model, we show that when label annotations are insufficiently fine-grained—as is often the case in real-world datasets—hidden stratification can naturally arise. For instance, an image classification task might be to classify birds

vs. frogs; if labels are only provided for these broad classes, they may fail to capture visually meaningful finer-grained, intra-class variation (e.g., "bird in flight" versus "bird in nest"). We show that in the setting of our generative model, standard training via empirical risk minimization (ERM) can result in arbitrarily poor performance on underrepresented subclasses.

Second, in Section 4 we use insights from this generative model to motivate GEORGE, a two-step procedure for alleviating hidden stratification by first *estimating* the subclass labels and then *exploiting* these estimates to train a robust classifier. To estimate subclass labels, we train a standard model on the task, and split each class (or "superclass," for clarity) into estimated subclasses via unsupervised clustering in the model's feature space. We then exploit these estimated subclasses by training a new model to optimize *worst-case* performance over all estimated subclasses using group distributionally robust optimization (GDRO [43]). In this way, our framework allows ML practitioners to automatically detect poorly-performing subclasses and improve performance on them, without needing to resort to expensive manual relabeling of the data.

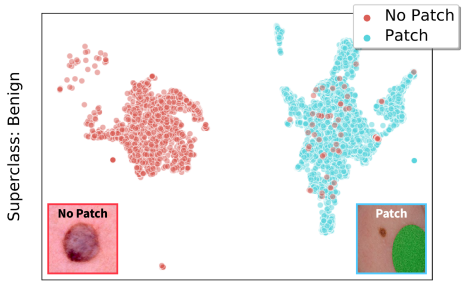


Figure 1: Benign class examples in the feature space of a model classfying skin lesions as benign or malignant. Benign examples containing a brightly colored patch (blue) and those without a patch (red) are separable in model feature space, even though the labels do not specify the presence of patches.

Third, in Section 5 we use our generative framework to prove that—under conditions on the data distribution and the quality of the recovered clusters—GEORGE can reduce the subclass performance gap, attaining the same asymptotic sample complexity rates as if the true subclass labels were known.

Fourth, in Section 6 we empirically validate the ability of GEORGE to both *measure* and *mitigate* hidden stratification on four image classification tasks, comprising both robustness benchmarks and real-world datasets. We demonstrate that the first step of GEORGE—training an ERM model and clustering the superclass features—often recovers clusters that align closely with true subclasses. We evaluate the ability of these clusters to measure the worst-case subclass (i.e., "robust") performance: on average, the gap between worst-case cluster performance and worst-case subclass performance is less than half the gap between overall and worst-case subclass performance, indicating that GEORGE enables more accurate measurement of robust performance. Next, we show that the second stage of GEORGE—retraining a robust model using cluster assignments as proxy subclass labels—reduces average worst-case subclass error rates by 22% on these datasets. For comparison, the state-of-the-art "oracle" GDRO method that *does* require subclass labels [43] reduces average worst-case subclass error rates by 51%. As an extension, we show that leveraging recent pretrained image embeddings [27] for clustering can substantially further improve the robust performance of GEORGE, in some cases to nearly match the performance of GDRO trained using the true subclass labels.

2 Background

2.1 Related Work

Our work builds upon prior work from three main areas: robust optimization, representation learning, and unsupervised clustering. We provide a more extensive discussion of related work in Appendix A.

Distributionally Robust Optimization Robustness and fairness is an active research area in machine learning [4, 21, 30, 26]. *Distributionally robust optimization* (DRO) attempts to guarantee good performance in the presence of distribution shift, e.g., from adversarial perturbations [49, 47] or evaluation on arbitrary subpopulations [16]. Because these notions of robustness can be pessimistic [23], others investigate *group* DRO (GDRO), which optimizes worst-case performance over a known set of subgroups [23, 43]. A major obstacle to applying GDRO methods in practice is that subgroup labels are often unavailable; in our work,

we aim to address this issue in the classification setting.

Representation Learning & Clustering Our approach relies on estimating unknown subclass labels by clustering a feature representation of the data. Techniques for learning semantically useful image features include autoencoder-based methods [32, 46], the use of unsupervised auxiliary tasks [2, 9], and pretraining on massive datasets [27]. Such features may be used for unsupervised identification of classes, either using clustering techniques [6] or an end-to-end approach [25, 18]. It has also been observed that when a model is trained on coarse-grained class labels, the data *within each class* can often be separated into distinct clusters in model feature space (e.g., [36]). While we primarily focus on the latter approach, we also evaluate the utility of pretrained embeddings as a source of features for clustering.

2.2 Problem Setup

We are given n datapoints $x_1, \ldots, x_n \in \mathcal{X}$ and associated *superclass* labels $y_1, \ldots, y_n \in \{1, \ldots, B\}$. In addition, each datapoint x_i is associated with a latent (unobserved) *subclass* label $z_i \in \{1, \ldots, C\}$. We assume that $\{1, \ldots, C\}$ is partitioned into disjoint sets $S_1, \ldots, S_B$ such that if $z_i \in S_b$, then $y_i = b$; in other words, the subclass label z_i determines the superclass label y_i. Let S_b denote the set of all subclasses comprising superclass b, and $S(c)$ denote the superclass corresponding to subclass c.

Our goal is to classify examples from $\mathcal{X}$ into their correct *superclass*. Given a function class $\mathcal{F}$, it is typical to seek a classifier $f \in \mathcal{F}$ that maximizes overall population accuracy:

$$\operatorname*{argmax}_{f \in \mathcal{F}} \mathbb{E}_{(x,y)} \left[\mathbf{1}(f(x) = y) \right] \ . \tag{1}$$

By contrast, we seek to maximize the *robust accuracy*, defined as the *worst-case* expected accuracy over all subclasses:

$$\operatorname*{argmax}_{f \in \mathcal{F}} \min_{c \in \{1,\ldots,C\}} \mathbb{E}_{(x,y)|z=c} \left[\mathbf{1}(f(x) = y) \right] \ . \tag{2}$$

Note that y is fixed conditional on the value of z. As we cannot directly optimize the population accuracy, we select a surrogate loss function ℓ and attempt to minimize this loss over the training data. For instance, the standard ERM approach to approximate (1) minimizes the empirical risk (loss) $R(f)$:

$$\operatorname*{argmin}_{f \in \mathcal{F}} \left\{ R(f) := \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i) \right\} \ . \tag{3}$$

To approximate (2), if we knew $z_1, ..., z_n$ we could minimize the *worst-case per-subclass training loss* by solving:

$$\operatorname*{argmin}_{f \in \mathcal{F}} \left\{ R_{\text{robust}}(f) := \max_{c \in \{1,\ldots,C\}} \frac{1}{n_c} \sum_{i=1}^{n} \mathbf{1}(z_i = c) \ell(f(x_i), y_i) \right\}, \tag{4}$$

where $n_c = \sum_{i=1}^{n} \mathbf{1}(z_i = c)$ is the number of training examples from subclass c. $R_{\text{robust}}(f)$ is the "robust loss" achieved by f. Our goal is to learn a model $\tilde{f} \in \mathcal{F}$ such that $R_{\text{robust}}(\tilde{f}) - \min_{f \in \mathcal{F}} (R_{\text{robust}}(f))$ is small with high probability. When the z_i's are known, Eq. (4) can be tractably optimized using GDRO [23, 43]. However, we do *not* assume access to the z_i's; we seek to approximately minimize R_{robust} without knowledge of the subclass labels.

3 Modeling Hidden Stratification

In Section 3.1, we introduce a generative model of the data labeling process. In Section 3.2, we use this model to explain how hidden stratification can occur, and show that in the setting of this model ERM can attain arbitrarily poor robust risk compared to DRO.

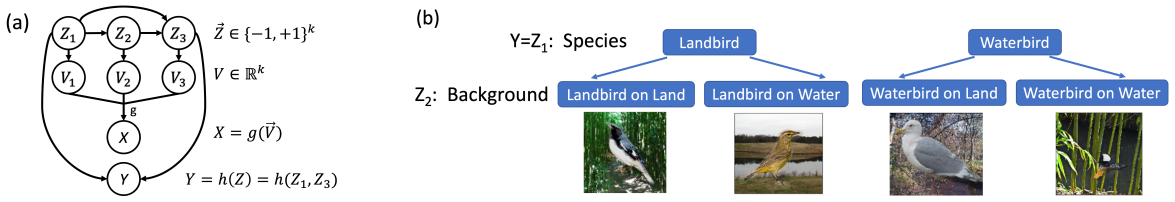


Figure 2: (a) Generative model of hidden stratification: attributes Z determine features $\vec{V}$ and labels Y; mapping g transforms $\vec{V}$ to yield observed data X. (b) On the Waterbirds dataset [43], attributes (Z_1, Z_2) denote species and background type respectively; the label Y is the species type.

3.1 A Model of the Data Generating and Labeling Process

In real datasets, individual datapoints are typically described by multiple different attributes, yet often only a subset of these are captured by the class labels. For example, a dataset might consist of images labeled "cat" or "dog." These coarse class labels may not capture other salient attributes (color, size, breed, etc.); these attributes can be interpreted as latent variables representing different subclasses.

We model this phenomenon with a hierarchical data generation process. First, a binary vector $\vec{Z} \in \{-1, +1\}^k$ is sampled from a distribution $p(\vec{Z})$. Each entry Z_i is an attribute, while each unique value of $\vec{Z}$ represents a different subclass. Then, a latent "feature vector" $\vec{V} \in \mathbb{R}^k$ is sampled from a distribution conditioned on $\vec{Z}$. Specifically, when conditioned on Z_i, each individual feature V_i is Gaussian and independent of the Z_j's with $j > i$. Finally, the datapoint $X \in \mathcal{X}$ is determined by the latent features $\vec{V}$ via a fixed map $g : \mathbb{R}^k \to \mathcal{X}$. Meanwhile, the superclass label Y is equal to $h(\vec{Z})$, where h is a fixed discrete-valued function. In particular, h may only depend on a subset of the Z_i attributes; the Z_i's which do not influence the label Y correspond to hidden subclasses. X, Y are observed, while $\vec{V}, \vec{Z}$ are not. Figure 2a illustrates this generative process; Figure 2b presents an analogue on the Waterbirds dataset [43].

A key assumption is that the subclasses are "meaningful" in some sense, rather than just arbitrary groups of datapoints. Thus, rather than attempting to enforce good performance on *all possible subsets* of the data, we assume some meaningful structure on the subclass data distributions. We model this via the Gaussian assumption on $p(V_i|\vec{Z})$, which is similar to that often made for the latent space of GANs [5]. Consequently, the data distribution is a mixture of Gaussians in "feature space," which facilitates further theoretical analysis (Section 5). Our generative model also bears similarity to that of [23], who use a hierarchical data-generation model to analyze the behavior of DRO methods in the presence of distribution shift.

3.2 What Causes Hidden Stratification, and When Can It Be Fixed?

We now use our generative model to help understand why hidden stratification can occur, and present a simple example in which ERM is provably suboptimal in terms of the robust risk.

We distinguish between two main causes of hidden stratification: *inherent hardness* and *dataset imbalance*. First, certain subclasses are "inherently harder" to classify because they are more similar to other superclasses. We define the inherent hardness of a task as the minimum attainable robust error; inherent hardness thus lower bounds the worst-case subclass error of any model. See Appendix D for more discussion.

Second, imbalance in subclass sizes can cause ERM to underserve rare subclasses, since it optimizes for average-case performance. We provide a simple concrete example (3.1) below. Unlike inherent hardness, robust performance gaps arising from dataset imbalances *can be resolved* if subclass labels are known, by using these labels to minimize Eq. (4) via GDRO.

Example 3.1 Figure 3 depicts an example distribution generated by the model in Section 3.1. In this example, the binary attribute vector $\vec{Z}$ has dimension 2, i.e., $\vec{Z} = (Z_1, Z_2)$, while only Z_2 determines the superclass label Y, i.e., $Y = Z_2$. The latent attribute Z_1 induces two subclasses in each superclass, each distributed as a different Gaussian in feature space, with mixture proportions α and $1 - \alpha$ respectively. For linear models with regularized logistic loss, as the proportion α of the rare subclasses goes to 0, the worst-case

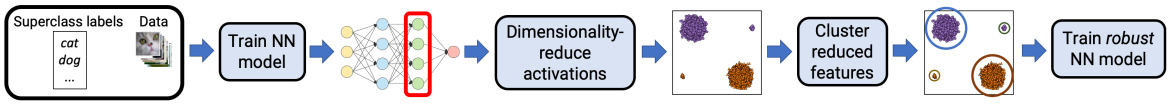


Figure 4: Schematic describing GEORGE. The inputs are the datapoints and superclass labels. First, a model is trained with ERM on the superclass classification task. The activations of the penultimate layer are then dimensionality-reduced, and clustering is applied to the resulting features to obtain estimated subclasses. Finally, a new model is trained using these clusters as groups for GDRO.

subclass accuracy of ERM is only $O(\alpha)$, while that of GDRO is $1 - O(\alpha)$. (See Appendix D.1 for the specific parameters of the per-subclass distributions in this example and a proof of the claim.)

Example 3.1 illustrates that when the dataset is imbalanced—i.e., the distribution of the underlying attributes $\vec{Z}$ is highly nonuniform—knowledge of subclass labels can improve robust performance. We thus ask: *how well can we estimate subclass labels if they are not provided*? In the extreme, if two subclasses of a superclass have the same distribution in feature space, we cannot distinguish them. However, the model must then perform the same on each subclass, since its prediction is a fixed function of the features! Conversely, if one subclass has higher average error, it must lie "further across" the decision boundary, meaning that the two subclasses must be separable to some degree; the larger the accuracy gap, the more separable the subclasses are. We formalize this in Appendix D.3.

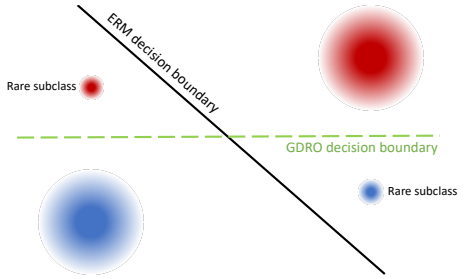


Figure 3: As $\alpha \to 0$ in Example 3.1, top-left & lower-right subclasses get rarer and are misclassified by ERM (black boundary), whereas GDRO learns the optimal robust boundary (green) to classify red vs. blue superclasses.

4 GEORGE: A Framework for Mitigating Hidden Stratification

Inspired by the insights of Section 3, we propose GEORGE, an algorithm to mitigate hidden stratification. A schematic overview of GEORGE is provided in Figure 4.

Under the generative model of Section 3.1, each subclass is described by a different Gaussian in latent feature space. This suggests that a natural approach to *identify* the subclasses is to transform the data into feature space, and then cluster the data into estimated subclasses. However, this feature space is unknown. To obtain a surrogate for this feature space, we leverage the empirical observation that feature representations of deep neural networks trained on a *superclass* task can carry information about unlabeled *subclasses* [36]. Next, to *improve performance* on these estimated subclasses, we minimize the maximum *per-cluster* average loss, by using the clusters as groups in the GDRO objective [43]. We provide more details below, and pseudocode in Appendix B (Algorithm 1).

4.1 Step 1: Estimating Approximate Subclass Labels

In the first step of GEORGE, we train an ERM model on the superclass task and cluster the feature representations of each superclass to generate proxy subclass labels. Formally, we train a deep neural network $L \circ f_\theta$ to predict the superclass labels, where $f_\theta : \mathcal{X} \to \mathbb{R}^d$ is a parametrized "featurizer" and $L : \mathbb{R}^d \to \Delta^B$ outputs classification logits. We then cluster the features output by f_θ for the data of each superclass into k clusters, where k is chosen automatically. To each datapoint x_i in the training and validation sets, we associate its cluster assignment $\tilde{z}_i \in \{1, \ldots, k\}$. We use the $\tilde{z}_i$'s as surrogates for the true subclass labels z_i.

4.1.1 Clustering Details

In practice, we apply UMAP dimensionality reduction [33] before clustering, as we find it improves results (Appendix B). Additionally, based on the insight of Section 3.2 that subclasses with high loss differences are more separable, we also use the loss component (i.e., the component of the activation vector orthogonal to the decision boundary) as an alternative representation.

We first tried using standard clustering methods (such as k-means and Gaussian mixture model clustering) in our work. By visual inspection, we found that these methods often failed to capture smaller clusters, even if they were well-separated. However, missing small clusters like this is problematic for GEORGE, since these small clusters frequently correspond to rare, low-performing subclasses. Additionally, these methods require specification of k. We apply *over-clustering* (clustering using a larger k) to remedy this problem in an efficient manner. Naive overclustering also has drawbacks as it still requires manual specification of k, and if k is set too large, several clusters can be spurious and result in overly pessimistic and unstable measurements of robust performance (as we explore in Appendix C.2.5). Thus, we develop a fully automated criterion based on the commonly used Silhouette (SIL) criterion [42] to search for the number of clusters k, over-cluster to find smaller clusters that were missed, and filter out the spurious overclusters. Empirically, our clustering approach significantly improves performance over "vanilla" clustering; we hope that it may be of independent interest as well. We describe our procedures in more detail in Appendix B.

k and other clustering and dimensionality reduction hyperparameters are selected automatically based on an unsupervised SIL criterion [42] as described further in Appendix B.

4.2 Step 2: Exploiting Approximate Subclass Labels

In the second step of GEORGE, we use the GDRO algorithm from [43] and our estimated subclass labels $\tilde{z}_i$ to train a new classifier with better worst-case performance on the estimated subclasses. Given data $\{(x_i, y_i, t_i)\}_{i=1}^n$ and loss function ℓ, GDRO minimizes $\max_{t \in \mathcal{T}} \mathbb{E}_{x,y \sim \hat{P}_t} [\ell((L \circ f_\theta)(x), y)]$ with respect to parameters (L, θ), where $\mathcal{T}$ is the discrete set of groups and $\hat{P}_t$ is the empirical distribution of examples from group t. This coincides with the true objective (4) when the true subclass labels z_i are used as the group labels t_i. In our case, we use the cluster assignments $\tilde{z}_i$ as the group labels instead, i.e., minimize $\max_{1 \le \tilde{z} \le k} \mathbb{E}_{x,y \sim \hat{P}_{\tilde{z}}} [\ell(L \circ f_\theta)(x), y)]$.[1]
Similarly, using the clusters fit on the training set, we generate cluster assignments for the validation set points and use these to compute the validation worst-case per-cluster performance. GEORGE uses this metric, rather than overall validation performance, to select the best model checkpoint over the training trajectory.

5 Analysis of GEORGE

We now analyze a simple mixture model data distribution, based on the generative model presented in Section 3.1. We show that in this setting, unlike ERM, GEORGE converges to the optimal robust risk at the same sample complexity rate as GDRO when it is able to recover the true latent features $\vec{Z}$. Specifically, Example 3.1 shows that the robust risk of ERM can be arbitrarily worse than that of GDRO for data generated according to the generative model in Section 3.1. By contrast, if the subclass labels estimated by GEORGE are sufficiently accurate, then the objective minimized in Step 2 of GEORGE well approximates the true GDRO objective (4). In Theorem 1, we use this to show that, when each subclass is described by a different Gaussian in feature space, GEORGE achieves the same optimal asymptotic sample complexity rates as GDRO trained with true subclass labels. We sketch the argument below; full proofs are deferred to Appendix D.

First, suppose we could compute the true data distribution $\mathcal{P}(x, y, z)$. Our goal is to minimize the maximum

[1] In Appendix D, we present an extension to the GDRO algorithm of [43] to handle the case where the group assignments $\tilde{z}_i$ can be probabilistic labels in Δ^k, instead of hard labels in $\{1, \ldots, k\}$.

per-subclass training loss by solving Eq. (4). Even with infinite data, we cannot estimate the *individual* z_i's to arbitrary accuracy, so we cannot directly compute the objective in (4). However, we can estimate the per-subclass losses as follows: for each training example (x_i, y_i), we use $\mathcal{P}$ to compute the probability that it comes from subclass c, and use that to weight the loss corresponding to that example. In Lemma 1, we show that when the training data is randomly sampled from $\mathcal{P}$, this yields an unbiased estimate of the average per-subclass empirical risk.

Lemma 1. *Let R_c be the sample average loss of examples in subclass c. Let $w(x, c) := \frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))}$. Let $\tilde{R}_c$ be the sample average of $w(x_i, c)\ell(f(x_i), y_i)$ over all examples x_i with* superclass *label $y_i = S(c)$. Then $\tilde{R}_c$ is an unbiased estimate of R_c, and their difference converges to 0 at $O(1/\sqrt{n})$.*

In practice, we do not have access to the true distribution $\mathcal{P}$, so we estimate it with $\hat{\mathcal{P}}$, computed from data. Thus, the weights $w(x, c)$ are replaced by weights $\hat{w}(x, c)$ estimated from $\hat{\mathcal{P}}$, leading to an estimate $\hat{R}_c$ of the quantity $\tilde{R}_c$ defined in Lemma 1. Nevertheless, if we can bound the total variation estimation error of $\hat{\mathcal{P}}$, we can use this to bound the error in this loss estimate, as shown in Lemma 2 (Appendix D). In Theorem 1, we leverage Lemma 2 and recent results on learning Gaussian mixtures [3] to show that, when each subclass is described by a different Gaussian, $\hat{\mathcal{P}}$ can be estimated well enough so that the minimizer of the perturbed robust loss converges to the minimizer of the true robust loss at the optimal sample complexity rate.

Theorem 1. *Let $\hat{R}_{robust} := \max_c \hat{R}_c$. Suppose ℓ, f are Lipschitz, f has bounded parameters, and $\mathcal{P}(x|z = c)$ is Gaussian and unique for each subclass c. Then, if we estimate $\hat{\mathcal{P}}$ using the algorithm from [3], $\hat{f} := \min_{f \in \mathcal{F}} \hat{R}_{robust}(f)$ satisfies $R_{robust}(\hat{f}) - \min_{f \in \mathcal{F}} R_{robust}(f) \leq \tilde{O}(\sqrt{1/n})$ w.h.p.*

Theorem 1 implies that if each subclass is Gaussian in feature space, and we have access to this feature space (i.e., we can invert the mapping g from features $\vec{Z}$ to data X), we can cluster the features to estimate $\hat{\mathcal{P}}$, and the robust generalization performance of the model that minimizes the resulting perturbed training loss $\hat{R}_{\text{robust}}$ scales the same as does that of the minimizer of the true robust training loss R_{robust}, in terms of the amount of data required. This underscores the importance of recovering a "good" feature space; empirically, we show in Appendix C that the choice of model architecture can indeed dramatically impact the model feature space and thus the ability to recover subclasses.

6 Experiments

We empirically validate that GEORGE can mitigate hidden stratification across four datasets. In Section 6.2, we show that when subclass labels are unavailable, GEORGE improves robust performance over standard training methods. In Section 6.3, we analyze the clusters returned by GEORGE to understand the reasons for this improvement; we confirm that GEORGE identifies clusters that correspond to poorly-performing subclasses, which enables accurate measurement of robust performance. In Section 6.4, we ablate the contributions that GEORGE's robust training objective and GEORGE's improved measurement of validation robust performance each make to the performance gains of GEORGE. Finally, in Section 6.5, we evaluate the use of recent pretrained image embeddings [27] as a source of features for GEORGE, and find that this further improves performance of GEORGE on some applications. Additional details on datasets, model architectures, and experimental procedures are provided in Appendix B.

6.1 Datasets

Waterbirds Waterbirds, a robustness benchmark introduced to evaluate GDRO in [43], contains images of land-bird and water-bird species on either land or water backgrounds. The task is to classify images as "land-bird" or "water-bird"; however, 95% of land (water)-birds are on land (water) backgrounds, causing ERM to frequently misclassify both land-birds on water and water-birds on land.

Method	Requires Subclass Labels?	Metric Type	Waterbirds	U-MNIST	ISIC		CelebA
					Non-patch	Histopath.	
ERM	✗	Robust	63.3(±1.6)	93.9(±0.6)	**.922**(±.003)	.875(±.005)	40.3(±2.3)
		Overall	97.3(±0.1)	98.2(±0.1)	.957(±.002)		95.7(±0.0)
George (ours)	✗	Robust	**76.2**(±2.0)	**95.7**(±0.6)	.912(±.005)	**.876**(±.006)	**53.7**(±1.3)
		Overall	95.7(±0.5)	97.9(±0.2)	.927(±.008)		94.6(±0.2)
Subclass-GDRO	✓	Robust	90.7(±0.4)	96.8(±0.4)	.923(±.003)	.875(±.004)	89.3(±0.9)
		Overall	92.7(±0.4)	98.0(±0.3)	.933(±.005)		92.8(±0.1)

Table 1: Robust and overall performance for ERM, George, and subclass-GDRO (i.e., GDRO with true subclass labels). Performance metric is accuracy for all datasets but ISIC, which uses AUROC. Bolded values are best between ERM and George, which do not require subclass labels. Sub-columns for ISIC represent two different definitions of the ISIC subclasses; see Section 6.3.

Undersampled MNIST (U-MNIST) We design U-MNIST as a modified version of MNIST [28], where the task is to classify digits as '<5' and '≥ 5' (digits 0-9 are the subclasses). In addition, we remove 95% of '8's; due to its rarity, it is challenging for ERM to perform well on the '8' subclass.

CelebA CelebA is a common face classification dataset also used as a robustness benchmark in [43]. The task is to classify faces as "blond" or "not blond." Because only 6% of blond faces are male, ERM performs poorly on this rare subclass.

ISIC The ISIC skin cancer dataset [12] is a public real-world dataset for classifying skin lesions as "malignant" or "benign." 48% of benign images contain a colored patch. Of the non-patch examples, 49% required histopathology (a biopsy) to diagnose. We report AUROC for ISIC, as is standard [41].

6.2 End-to-End Results

We first show that George substantially improves the worst-case subclass accuracy, while modestly affecting overall accuracy. (Recall that we refer to worst-case subclass accuracy as "robust accuracy" [Eq. (2)].) We train models on each dataset in Section 6.1 using (a) ERM, (b) George, and (c) GDRO with true subclass labels ("Subclass-GDRO"), and report both robust and overall performance metrics in Table 1. Compared to ERM, training with George substantially improves robust accuracy, reducing the gap between the robust error of the ERM model and that of the subclass-GDRO model—despite the fact that George does not require subclass labels. In Appendix C, we show that George also outperforms other subclass-agnostic baselines, such as GDRO trained using the *superclasses* as groups.

On Waterbirds, U-MNIST, and CelebA, George significantly improves worst-case subclass accuracy over ERM. On ISIC, all methods perform similarly in terms of both AUROC on the non-patch subclass and AUROC on the clinically meaningful histopathology subclass. On CelebA, although George improves upon ERM, it substantially underperforms subclass-GDRO. However, this gap can be closed when improved features are used: if we cluster pretrained BiT embeddings [27] rather than ERM features and use the resulting cluster assignments for the second stage of George, the robust accuracy improves to *nearly match* that of subclass-GDRO. We describe this experiment further in Section 6.5.

In terms of overall performance, ERM generally performs best (as it is designed to optimize for average-case performance), followed by George and then subclass-GDRO. However, this difference is generally much smaller in magnitude than the increase in robust performance.

6.3 Clustering Results

Step 1 of George is to train an ERM model and cluster the data of each superclass in its feature space. We analyze these clusters to better understand George's behavior. First, in Section 6.3.1 we show that

GEORGE finds clusters that align well with poorly-performing human-labeled subclasses. This helps explain why the second step of GEORGE, running GDRO using the cluster assignments as groups, improves performance on these subclasses (as demonstrated in Section 6.2). Next, in Section 6.3.2 we show that GEORGE can discover meaningful subclasses that were not labeled by human annotators. Finally, in Section 6.3.3 we show that the worst-case performance measured on the clusters returned by GEORGE is a good approximation of the true robust performance.

6.3.1 Subclass Recovery

We evaluate the ability of GEORGE to identify clusters that correspond to the true subclasses. We focus on identification of *poorly-performing* subclasses, as these determine robust performance. In Table 2, we compute the precision and recall of the cluster returned by GEORGE that most closely aligns with each given subclass. *Precision* is the fraction of cluster examples with that subclass label; *recall* is the fraction of subclass examples assigned to the cluster. For each poorly-performing subclass, GEORGE identifies a cluster with high recall and better-than-random precision.

We note that the lower recall on ISIC is because the no-patch subclass is often split into multiple clusters; in fact, this subclass is actually composed of two semantically distinct groups as discussed below. If these sub-clusters are combined into one, the precision and recall of the resulting cluster at identifying no-patch examples are > 0.99 and > 0.97 respectively.

6.3.2 Unlabeled Subclass Discovery

In addition to yielding clusters aligned with human-annotated subclasses, our procedure can identify semantically meaningful subclasses that were not specified in the human-provided schema. On U-MNIST, 60% of trials of GEORGE partition the "7" subclass into two subclusters, each containing stylistically different images (Figure 6c, Appendix C). On ISIC, 70% of GEORGE trials reveal distinct benign clusters within the no-patch subclass (see Figure 6g-i). In these trials, at least 77% of images in one of these no-patch clusters required histopathology (biopsy & pathologist referral), while such images made up <7% of each other cluster. In other words, the no-patch subclass split into "histopathology" and "non-histopathology" clusters, where the former datapoints were harder for clinicians to classify. We comment on the real-world importance of this result in the Broader Impacts section.

Task	Subclass	Subclass Prevalence	% of trials	Precision	Recall
U-MNIST	"8" digit	0.012	80	0.54	0.74
Waterbirds	Water-birds on land	0.05	100	0.19	0.91
Waterbirds	Land-birds on water	0.05	100	0.33	0.93
ISIC	No-patch	0.48	100	0.99	0.59
ISIC	Histopathology	0.23	70	0.77	0.92
CelebA	blond males	0.06	100	0.14	0.88
CelebA (w/BiT)	blond males	0.06	100	0.93	0.68

Table 2: Alignment of clusters with poorly-performing subclasses on the train set. We run Step 1 of GEORGE over multiple random seeds (i.e., train multiple ERM models and cluster their activations). In col. 4, we report the percentage of these trials with a cluster above the given precision and recall thresholds (cols. 5, 6) for identifying the subclass in col. 2. We report the proportion of training examples from that subclass within its superclass in col. 3.

6.3.3 Estimating Robust Accuracy

We show that the clusters returned by GEORGE enable improved measurement of worst-case subclass performance. Specifically, we measure the worst-case performance across any cluster returned by GEORGE (which we call the "cluster-robust" performance) and compare this to the true robust performance and the overall performance. We present results for both ERM and GEORGE in Table 3. In most cases, the cluster-robust performance is much closer to the true robust performance than the overall performance is. On ISIC,

cluster-robust performance even yields a better estimate of robust performance on the histopathology subclass than does performance on the patch/no-patch subclass. By comparing cluster-robust performance to overall performance, we can detect hidden stratification (and estimate its magnitude) without requiring subclass labels.

Method	Metric Type	Waterbirds	U-MNIST	ISIC Non-patch	ISIC Histopath.	CelebA
ERM	Robust	63.3(±1.6)	93.9(±0.6)	.922(±.003)	.875(±.005)	40.3(±2.3)
	Cluster-Robust	76.8(±1.4)	92.3(±2.5)	.893(±.013)		56.7(±2.5)
	Overall	97.3(±0.1)	98.2(±0.1)	.957(±.002)		95.7(±0.1)
GEORGE	Robust	76.2(±2.0)	95.7(±0.6)	.912(±.005)	.876(±.006)	53.7(±1.3)
	Cluster-Robust	78.3(±1.1)	93.5(±1.9)	.897(±.011)		70.8(±1.1)
	Overall	95.7(±0.5)	97.9(±0.2)	.927(±.008)		94.6(±0.2)

Table 3: Comparison of overall, cluster-robust, and robust performance.[2](Conventions as in Table 1.)

In addition, improvements in robust performance from GEORGE compared to ERM are accompanied by increases in cluster-robust performance; by comparing the cluster-robust performance of ERM and GEORGE, we can estimate how much GEORGE improves hidden stratification.

6.4 Effects of Validation Metric

GEORGE's improvement of robust performance has two potential explanations: (1) minimizing the cluster-robust training loss is a better surrogate objective for the true robust performance than minimizing the overall training loss, and (2) selecting the best model checkpoint based on validation cluster-robust performance is better than selecting based on overall validation performance. To decouple these two effects, in Table 4 we display the test robust performance for ERM and GEORGE when using the *true* robust validation performance as a checkpointing metric. This change generally improves performance for both methods, but GEORGE still significantly outperforms ERM on all datasets except ISIC. This shows that, for the goal of maximizing robust performance, the GDRO objective with cluster labels indeed performs better than the ERM objective.

In Table 4, we also display the effects of changing the frequencies of subclasses in the validation set. As described in Appendix B.2, by default we re-weight the validation and test sets of U-MNIST and Waterbirds so that the effective frequencies of each subclass are the same as they are in the training set. If we turn off this reweighting, the cluster-robust validation performance is a more accurate measure of the true robust performance, since the frequency of the underperforming subclass increases in the validation set. Using this unreweighted metric to select the best model checkpoint increases the true robust performance of GEORGE to 83.3%—an improvement of over **22** points compared to ERM checkpointed against average accuracy on the same unreweighted validation set. We stress that the true validation subclass labels are still assumed to be unknown in this experiment. Having training and validation sets with different distributions is realistic in many situations.

6.5 Extension: Leveraging Pretrained Embeddings

As an alternative to training an ERM model, we assess whether recent pretrained image embeddings (BiT, [27]) can provide better features for Step 1 of GEORGE. Specifically, we modify Step 1 of GEORGE to compute BiT embeddings for the datapoints, cluster the embeddings, and use these cluster assignments as estimated subclass labels in Step 2 of GEORGE. This modification (GEORGE-BiT) dramatically improves robust accuracy on CelebA to **87.3%** (±1.3%), nearly matching subclass-GDRO.[3]

[3]Overall accuracy drops somewhat to 91.5%.

Method	Checkpoint Metric	Waterbirds	U-MNIST	ISIC		CelebA
				Non-patch	Histopath.	
ERM	Acc.	63.3(±1.6)	93.9(±0.6)	.922(±.003)	.875(±.005)	40.3(±2.3)
	Unw. Acc.	60.7(±0.7)	94.2(±0.8)	-		-
	Robust Acc.	68.8(±0.9)	94.5(±0.9)	.924(±.003)	.880(±.004)	46.3(±2.1)
George	Cluster-Robust Acc.	76.2(±2.0)	95.7(±0.6)	.912(±.005)	.876(±.006)	53.7(±1.3)
	Unw. Cluster-Robust Acc.	83.3(±1.3)	95.7(±0.6)	-		-
	Robust Acc.	83.8(±1.0)	96.5(±0.2)	.915(±.004)	.877(±.006)	54.9(±1.9)

Table 4: Test robust performance for each method, where the "best" model checkpoint over the training trajectory is selected according to the listed metric on the validation set. "Unw." stands for unweighted, where we do not reweight the frequencies of different subclasses in the evaluation sets. This only applies to Waterbirds and U-MNIST, as the subclass proportions of ISIC and CelebA are roughly equal across splits. (For ISIC, the checkpoint metrics are AUROC rather than accuracy; in the two subcolumns we define the true robust performance as AUROC for the non-patch or histopathology subclass, respectively.)

The CelebA BiT clusters align much better with the true subclasses (cf. Table 2), which helps explain this improvement. Similarly, cluster-robust accuracy measured using the BiT clusters is much closer to the true robust accuracy: for the George-BiT model, average cluster-robust performance on the BiT clusters is $83.3 \pm 1.3\%$, and for ERM it is $33.9 \pm 2.5\%$ [compared to the true robust accuracy of 40.3%].

Despite its excellent performance on CelebA, the default George implementation outperforms George-BiT on the other datasets, suggesting that BiT is not a panacea: on these datasets, the task-specific information contained in the representation of the trained ERM model seems to be important for identifying meaningful clusters. See Appendix B.3.5 for additional evaluations and discussion. Extending Step 1 of George to enable automatically selecting between different representations (e.g., BiT vs. ERM) is a compelling future topic.

7 Conclusion

We propose George, a two-step approach for measuring and mitigating hidden stratification without requiring access to subclass labels. George's first step, clustering the features of an ERM model, identifies clusters that provide useful approximations of worst-case subclass performance. George's second step, using these cluster assignments as groups in GDRO, yields significant improvements in worst-case subclass performance. We analyze George in the context of a simple generative model, and show that under suitable assumptions George achieves the same asymptotic sample complexity rates as if we had access to true subclass labels. We empirically validate George on four datasets, and find evidence that it can reduce hidden stratification on real-world machine learning tasks. Interesting directions for future work include further exploring different ways to learn representations for the first stage of George, developing better unsupervised metrics to choose between representations and clustering methods, and characterizing when ERM learns representations that enable separation of subclasses.

Broader Impact

The potential real-world impact of George, the approach we present in this work, is that it would allow machine learning practitioners to both *measure* and *mitigate* hidden stratification without requiring any additional prior information. Concretely, this means that users would be able to leverage clusters identified in Step 1 of George to measure performance gaps between unlabeled subclasses, and that they would subsequently be able to reduce that subclass performance gap via Step 2 of George. We hope that George could serve as a drop-in replacement for standard ERM-based techniques in situations where ensuring good performance across many potentially unknown subclasses is important, as it is simple to implement and can be generically applied: all that is required to apply George to an existing model is (a) clustering within the representation space of a trained model and (b) retraining using a GDRO objective with the cluster assignments used as groups.

As an example of how George could be important for meaningful practical applications, we consider our results presented on the ISIC dataset in a real-world context. Naively, the overall AUROC on the ISIC dataset obtained using an ERM-trained model is 0.957, which suggests a high-performing model; however, our clustering (Step 1 of George) reveals that a large fraction of the benign images contain a "spurious" brightly colored patch, which makes them very easy to classify. The model performs substantially worse for cases without such a patch, and worse still on "more difficult" cases for which a clinician would also have required a histopathology examination to make a diagnosis. Thus, if deployed in practice with a target sensitivity value in mind, the appropriate way to set an operating point for this model is in fact *cluster-dependent*; if a single operating point were set using the aggregate ROC curve, the true sensitivity on the histopathology subclass would be substantially lower than intended. This means that even just *measuring* hidden stratification via Step 1 of George can provide crucial information that would help avoid spurious false negatives at test time—the worst type of error a medical screening application can make.

As shown in the paper, Step 2 of George can improve performance on underperforming subclasses. Our approach thus provides additional value via a simple retraining procedure that can reduce the amount of hidden stratification exhibited by the model. While our approach will certainly not provide substantial gains in every possible case—for instance, if performance gaps between subclasses are already minimal—we also do not expect it to cause substantial performance degradation. Indeed, even if the clusters returned by George are random groupings of points that do not align well with the true subclasses, we still expect a model trained to be robust across such groups to perform similarly to a standard ERM model (as in this case the average per-cluster losses are likely to be close to the overall loss on the superclass). This conclusion is empirically supported by the random-GDRO results of Appendix B, which are generally comparable to ERM.

In summary, we hope that George will have broader impacts by (a) enabling better measurement of hidden stratification via Step 1, even without knowledge of the subclasses, and (b) potentially improving performance on underserved subclasses with Step 2, with only modest additional effort required compared to normal training procedures (i.e., clustering + retraining with GDRO). In addition to medical imaging tasks such as ISIC, subclasses could represent a large number of important categories including race, gender, and others on which one would generally want to ensure good performance on all subclasses, rather than optimizing for average performance while underserving certain categories. If successful, George can help to detect and mitigate such important performance differences before models are deployed in practice. To support these potential impacts, we have released a complete implementation of our code,[4] with an easily usable PyTorch API. We look forward to engaging with the broader community to improve our work and deploy it on real-world applications.

Acknowledgments

We thank Arjun Desai, Pang Wei Koh, Shiori Sagawa, Zhaobin Kuang, Karan Goel, Avner May, Esther Rolf, and Yixuan Li for helpful discussions and feedback.

[4] https://github.com/HazyResearch/hidden-stratification/

We gratefully acknowledge the support of DARPA under Nos. FA86501827865 (SDH) and FA86501827882 (ASED); NIH under No. U54EB020405 (Mobilize), NSF under Nos. CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), and 1937301 (RTML); ONR under No. N000141712266 (Unifying Weak Supervision); the Moore Foundation, NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, the Okawa Foundation, American Family Insurance, Google Cloud, Swiss Re, Total, the HAI-AWS Cloud Credits for Research program, the Schlumberger Innovation Fellowship program, and members of the Stanford DAWN project: Facebook, Google, VMWare, and Ant Financial. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of DARPA, NIH, ONR, or the U.S. Government.

References

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

[2] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. A critical analysis of self-supervision, or what we can learn from a single image. In *International Conference on Learning Representations (ICLR)*, 2020.

[3] Hassan Ashtiani, Shai Ben-David, Nick Harvey, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Near-optimal sample complexity bounds for robust learning of gaussian mixtures via compression schemes. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[4] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019. `http://www.fairmlbook.org`.

[5] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. In *International Conference on Machine Learning (ICML)*, 2018.

[6] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 132–149, 2018.

[7] Krzysztof Chalupka, Pietro Perona, and Frederick Eberhardt. Visual causal feature learning. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

[8] Beidi Chen, Weiyang Liu, Zhiding Yu, Jan Kautz, Anshumali Shrivastava, Anshumali Shrivastava, Animesh Garg, and Anima Anandkumar. Angular visual hardness. In *International Conference on Machine Learning (ICML)*, 2020.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

[10] Vincent Chen, Sen Wu, Alexander J Ratner, Jen Weng, and Christopher Ré. Slice-based learning: A programming model for residual learning in critical data slices. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9392–9402, 2019.

[11] Sasank Chilamkurthy, Rohit Ghosh, Swetha Tanamala, Mustafa Biviji, Norbert G Campeau, Vasantha Kumar Venugopal, Vidur Mahajan, Pooja Rao, and Prashant Warier. Deep learning algorithms for detection of critical findings in head CT scans: a retrospective study. *Lancet*, 392(10162):2388–2396, December 2018.

[12] Noel Codella, Veronica Rotemberg, Philipp Tschandl, M Emre Celebi, Stephen Dusza, David Gutman, Brian Helba, Aadi Kalloo, Konstantinos Liopyris, Michael Marchetti, et al. Skin lesion analysis toward melanoma detection 2018: A challenge hosted by the international skin imaging collaboration (ISIC). *arXiv preprint arXiv:1902.03368*, 2019.

[13] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional Gaussians. *arXiv preprint arXiv:1810.08693*, 2018.

[14] Jian Dong, Qiang Chen, Jiashi Feng, Kui Jia, Zhongyang Huang, and Shuicheng Yan. Looking inside category: subcategory-aware object recognition. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(8):1322–1334, 2014.

[15] Jian Dong, Qiang Chen, Jiashi Feng, Kui Jia, Zhongyang Huang, and Shuicheng Yan. Looking inside category: Subcategory-aware object recognition. *Circuits and Systems for Video Technology, IEEE Transactions on*, 25:1322–1334, 08 2015. doi: 10.1109/TCSVT.2014.2355697.

[16] John C Duchi, Tatsunori Hashimoto, and Hongseok Namkoong. Distributionally robust losses against mixture covariate shifts. *arXiv preprint arXiv:2007.13982*, 2020.

[17] Jared A Dunnmon, Darvin Yi, Curtis P Langlotz, Christopher Ré, Daniel L Rubin, and Matthew P Lungren. Assessment of convolutional neural networks for automated classification of chest radiographs. *Radiology*, 290(2):537–544, February 2019.

[18] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Learning to classify images without labels. *arXiv preprint arXiv:2005.12320*, 2020.

[19] Varun Gulshan, Lily Peng, Marc Coram, Martin C Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, Ramasamy Kim, Rajiv Raman, Philip C Nelson, Jessica L Mega, and Dale R Webster. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410, December 2016.

[20] Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 8401–8409, 2019.

[21] Moritz Hardt, Eric Price, and Nati Srebro. Equality of opportunity in supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3315–3323, 2016.

[22] Minh Hoai and Andrew Zisserman. Discriminative sub-categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1666–1673, 2013.

[23] Weihua Hu, Gang Niu, Issei Sato, and Masashi Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning (ICML)*, 2018.

[24] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[25] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 9865–9874, 2019.

[26] Michael Kearns, Aaron Roth, and Saeed Sharifi-Malvajerdi. Average individual fairness: Algorithms, generalization and experiments. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[27] Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big transfer (BiT): General visual representation learning. *arXiv preprint arXiv:1912.11370*, 2020.

[28] Yann LeCun, Corinna Cortes, and CJ Burges. MNIST handwritten digit database. *ATT Labs [Online]. http://yann.lecun.com/exdb/mnist*, 2010.

[29] Percy Liang and Tengyu Ma. 229T course notes, 2019. URL `http://web.stanford.edu/class/cs229t/`.

[30] Zachary Lipton, Julian McAuley, and Alexandra Chouldechova. Does mitigating ML's impact disparity require treatment disparity? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8125–8135, 2018.

[31] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[32] Ryan McConville, Raul Santos-Rodriguez, Robert J Piechocki, and Ian Craddock. N2D: (not too) deep clustering via clustering the local manifold of an autoencoded embedding. *arXiv preprint arXiv:1908.05968*, 2019.

[33] Leland McInnes, John Healy, and James Melville. UMAP: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[34] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.

[35] Rafael Muller, Simon Kornblith, and Geoffrey Hinton. Subclass distillation. *arXiv preprint arXiv:2002.03936*, 2020.

[36] Luke Oakden-Rayner, Jared Dunnmon, Gustavo Carneiro, and Christopher Ré. Hidden stratification causes clinically meaningful failures in machine learning for medical imaging. In *Proceedings of the ACM Conference on Health, Inference, and Learning (CHIL)*, 2020.

[37] Neoklis Polyzotis, Steven Whang, Tim Klas Kraska, and Yeounoh Chung. Slice finder: Automated data slicing for model validation. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*, 2019.

[38] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *The VLDB Journal*, pages 1–22, 2019.

[39] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do CIFAR-10 classifiers generalize to CIFAR-10? *arXiv preprint arXiv:1806.00451*, 2018.

[40] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? In *International Conference on Machine Learning (ICML)*, 2019.

[41] Laura Rieger, Chandan Singh, W. James Murdoch, and Bin Yu. Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. *arXiv preprint arXiv:1909.13584*, 2019.

[42] Peter J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987.

[43] 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. In *International Conference on Learning Representations (ICLR)*, 2020.

[44] Bernhard Schölkopf. Causality for machine learning. *arXiv preprint arXiv:1911.10500*, 2019.

[45] Alon Scope, Michael A. Marchetti, Ashfaq A. Marghoob, Stephen W. Dusza, Alan C. Geller, Jaya M. Satagopan, Martin A. Weinstock, Marianne Berwick, and Allan C. Halpern. The study of nevi in children: Principles learned and implications for melanoma diagnosis. *Journal of the American Academy of Dermatology*, 75(4):813 – 823, 2016. ISSN 0190-9622. doi: https://doi.org/10.1016/j.jaad.2016.03.027. URL `http://www.sciencedirect.com/science/article/pii/S019096221630010X`.

[46] Ankita Shukla, Gullal Singh Cheema, and Saket Anand. Semi-supervised clustering with neural networks. *arXiv preprint arXiv:1806.01547*, 2018.

[47] Aman Sinha, Hongseok Namkoong, Riccardo Volpi, and John Duchi. Certifying some distributional robustness with principled adversarial training. In *International Conference on Learning Representations (ICLR)*, 2018.

[48] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4077–4087, 2017.

[49] Matthew Staib and Stefanie Jegelka. Distributionally robust deep learning as a generalization of adversarial training. In *NIPS Workshop on Machine Learning and Computer Security*, 2017.

[50] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 10506–10518, 2019.

[51] Pengtao Xie, Aarti Singh, and Eric P. Xing. Uncorrelation and evenness: a new diversity-promoting regularizer. In *International Conference on Machine Learning (ICML)*, 2017.

[52] Bangpeng Yao, Aditya Khosla, and Li Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1577–1584. IEEE, 2011.

Appendix

A Related Work

Our work builds on several active threads in the machine learning literature.

Hidden Stratification Our motivating problem is that of hidden stratification, wherein models trained on superclass labels exhibit highly variable performance on unlabeled subclasses [36]. This behavior has been observed in a variety of studies spanning both traditional computer vision [52, 39, 14, 22] and medical machine learning [36, 17, 19, 11]. Of note is the work of [39], who propose that the existence of "distribution shift" at the subclass level may substantially affect measures of test set performance for image classification models on CIFAR-10. They use a simple mixture model between an "easy" and a "hard" subclass to demonstrate how changes that would not be detectable at the superclass level could affect aggregate performance metrics. [8] extend these ideas by developing notions of visual hardness, and suggest that better loss function design would be useful for improving the performance of machine learning models on harder examples. [37] study how to automatically find large, interpretable underperforming data slices in structured datasets.

Our approach is also inspired by the literature on causality and machine learning, and in particular by the common assumption that the data provided for both training and evaluation are independent and identically distributed (IID) [44]. This is often untrue in real-world settings; in particular, classes in real-world datasets are often composed of multiple subclasses, and the proportions of these subclasses may change between training and evaluation settings—even if the overall class compositions are the same. Many of the guarantees from statistical learning theory break down in the presence of such non-IID data [7], suggesting that models trained using traditional Empirical Risk Minimization (ERM) are likely to be vulnerable to hidden stratification. This motivates the use of the maximum (worst-case) per-subclass risk, rather than the overall average risk, as the objective to be optimized.

Neural Representation Clustering The first stage of the technique we propose for addressing hidden stratification relies heavily on our ability to identify latent subclasses via unsupervised clustering of neural representations learned via ERM. This has been an area of substantial recent activity in machine learning, and has provided several important conclusions upon which we build in our work. The work of [32] and [46], for instance, demonstrate the utility of a simple autoencoded representation for performing unsupervised clustering in the feature space of a trained model. While the purpose of these works is often to show that deep clustering can be competitive with semi-supervised learning techniques, the mechanics of clustering in model feature space explored by these works are important for our present study. Indeed, we directly leverage the conclusion of [32] that Uniform Manifold Approximation and Projection (UMAP) [33] works well as a dimensionality reduction technique for deep clustering in the current study.

Further, the fact that work such as [20] directly uses neural representation clustering to estimate the presence of novel classes in a given dataset provides an empirical basis for our approach, which uses a model trained with ERM to approximately identify unlabeled subclasses within each superclass. Similarly, [25] demonstrate excellent semi-supervised image classification performance by maximizing mutual information between the class assignments of each pair of images. Their work demonstrates not only the utility of a clustering-style objective in image classification, but also suggests that overclustering – using more clusters than naturally exist in the data – can be beneficial for clustering deep feature representations in a manner that is helpful for semi-supervised classification.

A related, but different, approach is that of [14], who explicitly attempt to identify subcategories of classes via a graph and SVM-based "subcategory mining" framework in order to improve overall task performance. The subcategory mining algorithm is quite complicated and uses manually extracted features (rather than automatically learned features, e.g., from CNNs); in addition, this work is geared towards improving overall performance, rather than ensuring good performance on *all* subcategories. Nevertheless, it is an important piece of prior literature.

Distributionally Robust Optimization The second stage of our proposed approach depends on our ability to optimize the worst-case classification loss over existing subgroups, otherwise known as the Distributionally Robust Objective (DRO). This formulation draws a clear connection between our work and the literature on fairness in machine learning [4], which is at least partially concerned with ensuring that trained models do not disadvantage a particular group in practice. While there exist a wide variety of definitions for algorithmic fairness [21, 30, 26], the common idea that models should be optimized such that they respect various notions of fairness is closely related to the DRO formulation.

A multitude of recent studies have explored optimizing the DRO objective in slightly different contexts. [16], for instance, consider the general case of optimizing the worst-case loss over any possible subgroup of the data; while conceptually important, their results do not assume a learned feature representation, and the necessary assumptions are quite restrictive. Perhaps most relevant to the current work is the study of [43], who propose the group DRO algorithm for training classifiers with best worst-case subgroup performance. Crucially, this algorithm demonstrates improved worst-case subclass performance in cases where triplets (x, y, g) are known for every data point, with x is the input data, y is the true label, and g is a true subgroup label. While [43] present preliminary evidence that group DRO can work well in the presence of imperfect g, the efficacy of the algorithm in this setting remains functionally unexplored. We leverage the group DRO algorithm as an optimizer for solving the DRO objective with respect to our approximately identified subclasses.

Other relevant techniques include invariant risk minimization, which attempts to train classifiers that are optimal across data drawn from a mixture of distributions (i.e., a non-IID setting) [1]; methods from slice-based learning that learn feature representations optimized for ensuring high performance on specific subsets, or "slices" of the data [10]; mixture-of-experts models, which explicitly handle learning models for multiple different subsets of data [24]; and techniques for building robust classifiers via domain adaptation [50].

While our work is closely related to these directions, a major difference is that we handle the setting where the different subclasses (i.e., groups, environments, slices, etc.) are unidentified.

Representation Learning with Limited or Noisy Labels A final research thread that is closely related to the work presented here focuses on deep representation learning in the absence of ground truth labels. Our methods are similar in spirit to those from weak supervision [34, 38], which focuses on training models using noisy labels that are often provided programmatically. Our work can be seen as analyzing a new form of fine-grained weak supervision for DRO-style objectives, which is drawn from unsupervised clustering of an ERM representation. Another related line of work is representation learning for few-shot learning [48]; however, our work fundamentally differs in the sense that we assume no access to ground truth subclass labels.

Other methods aim to automatically learn classes via an iterative approach. An early work of this type is [6], which uses iterative clustering and ERM training to learn highly effective feature representations for image classification. More recently, [18] used a self-supervised task to learn semantically meaningful features, and then generate labels using an iteratively refining approach. Our work differs from these in that we do assume access to ground truth *superclass* labels—which provide much more information than in the fully-unlabeled setting—and use clustering *within each superclass* to generate approximate labels. In addition, our primary end goal is not accurate identification of the subclasses, but ensuring good worst-case performance among all subclasses.

Finally, other works aim to promote a notion of "diversity" among feature representations by adding different regularizers. In [51], such a regularizer was introduced in the context of latent space models, to better capture infrequently observed patterns and improve model expressiveness for a given size. More recently, [35] introduced a regularizer that aims to promote diversity of the predicted logits. They showed that this method could also lead to estimation of subclasses within a superclass, without requiring subclass labels. However, this work focused on improving *overall* performance, and specifically improvement of knowledge distillation; by contrast, our goal is to improve *robust* performance. Nevertheless, integrating these recent ideas into our work is an interesting avenue for future work, to potentially further improve the feature learning stage.

B Experimental Details

B.1 George Pseudocode

We provide pseudocode for George in Algorithm 1, to complement the detailed description of our methodology in Section 4.[5] Note that our model class $\mathcal{F}$ (as per the notation in Section 2.2) is a class of neural networks, composed of a "featurizer" module f_θ and a "linear classification head" L that takes the feature representation to a prediction.

Algorithm 1 "George"

Input: Data and superclass labels $(x, y) = \{(x_i, y_i)\}_{i=1}^n$; loss function $\ell(\cdot, \cdot)$; featurizer class $\mathcal{F}(\theta)$ parameterized by $\theta \in \mathbb{R}^p$, dimensionality reducer g, e.g., UMAP (default: identity)
Optional input: Pretrained featurizer f_θ

if featurizer f_θ provided **then**
  pass
else
  # train model [featurizer f_θ & linear classification head L] to minimize empirical risk, and save featurizer
  $f_\theta, L \leftarrow \underset{\theta' \in \mathbb{R}^p, L'}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_{i=1}^n \ell(L' \cdot f_{\theta'}(x_i), y_i) \right\}$
end if
# compute feature vectors
$\{v_i\}_{i=1}^n = g(f_\theta(x_i))$.
for $b = 1$ **to** B **do**
  # cluster features of each superclass
  $\{\hat{z}_i\} \leftarrow$ Get_Cluster_Labels($\{v_i : y_i = b\}$)
end for
# train final model to minimize maximum per-cluster risk
$f_{\hat{\theta}}, \hat{L} \leftarrow \underset{\theta' \in \mathbb{R}^p, L'}{\operatorname{argmin}} \left\{ \max_{c \in \{1, \ldots, C\}} \frac{1}{n_c} \sum_{i=1}^n \mathbf{1}(\hat{z}_i = c) \ell(L' \circ f_{\theta'}(x_i), y_i) \right\}$
return $(f_{\hat{\theta}}, \hat{L})$

Note that our framework is not constrained by the specific choice of clustering algorithm, dimensionality reduction algorithm, or robust optimization algorithm. While we use GDRO throughout this work, other training techniques that encourage good robust performance could be swapped in for GDRO during "Step 2" of George.

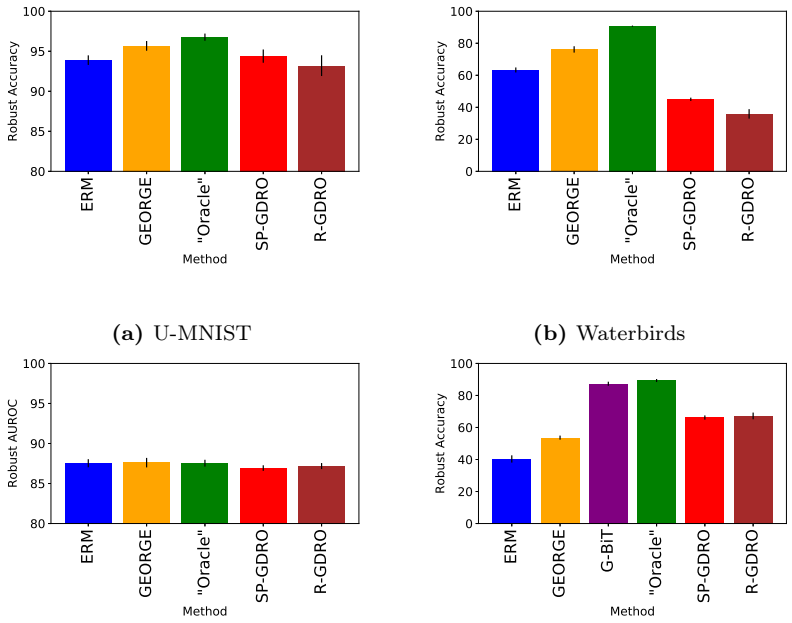
B.2 Dataset Details

Below, we describe the datasets used for evaluation in more detail. [We provide PyTorch dataloaders to support each one in our code.]

Each dataset contains labeled subclasses; although the George procedure does not use the subclass labels at any point, we use them to assess how well George (a) can estimate the subclass labels and (b) can estimate and improve worst-case subclass performance.

We remark that while we evaluate on binary classification tasks in this work, George can readily be applied in principle to tasks with any amount of superclasses and subclasses.

[5] We note that the final step of George—training a model to minimize the maximum per-cluster risk—can also be done when "soft" (probabilistic) cluster labels are given instead of hard assignments; see Appendix D.5.



(a) U-MNIST **(b)** Waterbirds

(c) ISIC (Histopathology) **(d)** CelebA

Figure 5: Performance of ERM, Superclass-GDRO (SP-GDRO), Random-GDRO (R-GDRO), GEORGE, and Subclass-GDRO (SBC-GDRO). We also show GEORGE with BiT embeddings (G-BiT) for CelebA; however, G-BiT performed significantly worse on the other datasets.

B.2.1 U-MNIST

Undersampled MNIST (U-MNIST) is a binary dataset that divides data from the standard MNIST dataset (which has 60,000 training points) into two superclasses: numbers less than five, and numbers greater than or equal to five. Crucially, the "8" subclass is subsampled at 5% of its usual frequency in MNIST. The rarity of the "8" subclass makes this task much more challenging than the default MNIST task, in terms of robust performance. We use data drawn from the original MNIST validation set as our test set; we create a separate validation set of 12,000 points by sampling from the MNIST training set, and use the remainder for training.

On the validation (and test) sets, we do not actually undersample the 8's, as this would leave only 50-60 "8" examples; instead, we downweight these examples when computing validation/test accuracies and losses, to mimic the rarity that would be induced by actually undersampling but still allow for more stable accuracy measurements.

B.2.2 Waterbirds

The Waterbirds dataset (4,795 training points) used in this work was introduced in [43]. Similar to our approach for U-MNIST, Sagawa et al. [43] create more balanced validation and test sets to allow for stable measurements, and downweight the examples from rare subclasses during evaluation (as we do for U-MNIST as well); we follow the same procedure.

B.2.3 ISIC

The dataset from the International Skin Imaging Collaboration (ISIC) website is, at time of writing, comprised of 23,906 images and their corresponding metadata [12]. We extract the ISIC dataset directly from the site's image archive, which is accessible through a public API.[6] We only use images whose metadata explicitly

[6]https://isic-archive.com/api/v1/

describe them as "benign" or "malignant." We use these descriptors in order to formulate the problem as a binary classification task that classifies images as either normal or abnormal. Other possible descriptors that exist in the image metadata (which we filter out) include "indeterminate," "indeterminate/benign," "indeterminate/malignant," or no description. We created pre-set training, validation, and test splits from these images by randomly assigning 80% of examples to the training set, 10% to the validation set, and 10% to the test set.

We derive true subclass information from the image metadata. In particular, we observed that an image belongs in the benign patch subclass if and only if it is an image from the SONIC data repository [45]. As is detailed in the paper, we are retroactively able to identify the histopathology subclass through analysis of the diagnosis confirmation type of each image. Images in the histopathology subclass were explicitly mentioned as such—other possible diagnosis confirmation types include "single image expert consensus," "serial imaging showing no change," "confocal microscopy with consensus dermoscopy," or no confirmation type.

B.2.4 CelebA

The CelebA dataset [31] is a standard face classification dataset containing over 200,000 examples ($\approx 163,000$ train) of celebrity faces, each annotated with 40 different attributes. The images contain a wide variety of poses, backgrounds, and other variations. The task is to classify faces as "blond" or "not blond," as in [43]. We use the standard (pre-set) train/validation/test splits for this task.

B.3 Methods

B.3.1 Result Reporting

For each dataset, we perform ten separate trials of each method with different random seeds. The exception is CelebA, on which we perform five trials instead due to the larger dataset size. In all result tables, '$X \pm Y$' intervals represents a 95% confidence interval, where X is the mean of the per-trial results and Y is the interval half-width, calculated as standard deviation times 1.96 divided by the square root of the number of trials. Similarly, in all plots, error bars denote 95% confidence intervals computed the same way.

B.3.2 Baselines

In addition to ERM, we run two additional baseline methods: superclass-GDRO and random-GDRO. Superclass-GDRO minimizes the maximum loss over each *superclass*, i.e., runs GDRO using the superclasses as groups. Since we assume knowledge of the training superclass labels, this does not require additional information at training time. Random-GDRO runs GDRO using *randomly chosen* groups within each superclass, where the groups are chosen to have the same sizes as the true subclasses. Since we do not assume the subclass sizes are known, this is not a method that would be useful in practice; rather, it helps highlight the difference between running GDRO with labels that do not align well with the true subclasses, and running GDRO with labels that do. Results on each dataset are presented in Figure 5.

B.3.3 ERM Training Details

The first stage of our procedure is to train a model for each application using ERM. The activations of the resulting model are clustered and used in the second stage of our procedure. Inspired by the results of [43], we explored using either a standard ERM model or an ERM model with high regularization for this stage, selecting between the two based on the quality of the resulting clustering as measured by the Silhouette score (an unsupervised metric). Below, we detail the ERM hyperparameter settings for each dataset.

U-MNIST Our U-MNIST model is a simple 4-layer CNN, based on a publicly available LeNet5 implementation;[7] based on this implementation, we fix the learning rate at 2e-3 and use the Adam optimizer. Each model is trained for 100 epochs. Because the original implementation does not specify a weight decay, we search over weight decay values of $[10^{-3}, 10^{-4}, 10^{-5}]$, and choose the setting with highest average validation accuracy over three trials with different random seeds. Our final hyperparameters are recorded in Table 5.

Waterbirds Our Waterbirds model uses the `torchvision` implementation of a 50-layer Residual Network (ResNet50), initialized with pretrained weights from ImageNet (as done in [43]). We use hyperparameters reported by [43]: weight decay of 1e-4, learning rate of 1e-3, SGD with momentum 0.9, and 300 epochs.

CelebA Our CelebA model also uses a `torchvision` pretrained ResNet50, as done in [43]. We use the hyperparameters reported by [43]: weight decay of 1e-4, SGD with momentum 0.9, and 50 epochs. However, we train on 4 GPUs instead of 1. (This change does not substantially affect the results; our ERM and subclass-GDRO results are similar to those reported in [43].)

ISIC Our ISIC model also uses a `torchvision` pretrained ResNet50. Models were trained for 20 epochs using SGD with momentum 0.9 (as done in [41]). Because these hyperparameters were unavailable in the literature for this architecture and task, we grid searched over weight decay values in [0.01, 0.001, 0.0001] and learning rates in [0.0005, 0.001, 0.005], selecting the values that maximize the overall AUROC on the validation set, averaged over three trials per hyperparameter setting.

Rather than measuring accuracy for ISIC, we use the AUROC (area under the receiver operating characteristic curve), as is standard on this task [41] and other medical imaging tasks [36]. The specific metric of interest is the worst *per-benign-subclass* AUROC for classifying between that subclass and the malignant superclass (e.g., benign no-patch vs. malignant AUROC). Typically, models designed to attain high AUROC are trained by minimizing the empirical risk as usual. For our *robust* models, we instead minimize $\max_{c \in \text{benign}} \left\{ \frac{1}{n_c + n_{\text{malignant}}} \sum_x \mathbf{1}(z_i = c \text{ OR } y_i = \text{malignant}) \ell(x_i, y_i; \theta) \right\}$ - in other words, the maximum over all benign subclasses of the "modified" empirical risk where all other benign subclasses are ignored. We do this because the worst-case loss over any *benign or malignant* subclass is not necessarily a good proxy for the worst-case per-*benign*-subclass AUROC. Due to the dataset imbalance (many fewer malignant than benign images), standard ERM models attain 100% accuracy on the benign superclass and much lower accuracy (and higher loss) on the malignant superclass. By contrast, in practice a classification threshold is typically selected corresponding to a target *sensitivity* value.

B.3.4 Clustering Details

We apply a consistent clustering procedure to each dataset, which is designed to encourage discovery of clusters of varied sizes, while still being computationally efficient. We emphasize that while the clustering procedure outlined below yields adequate end-to-end results on our datasets, optimizing this part of the GEORGE procedure represents a clear avenue for future work. In particular, we use the Silhouette score as a metric to select between feature representations and number of clusters; while this is a serviceable heuristic, it has several flaws (and in the case of BiT embeddings, misleadingly suggests that they are not a suitable representation due to their low Silhouette score).

1. *Dimensionality Reduction*: As recommended by [32], we use UMAP for dimensionality reduction before clustering; clustering is faster when the data is low-dimensional, and we find that UMAP also typically improves the results. As an alternative to UMAP, we also use the component of the representation that is orthogonal to the decision boundary, which we refer to as the "loss component," as a single-dimensional representation; this can improve clustering on datasets, especially when performance

[7] https://github.com/activatedgeek/LeNet-5

on certain subclasses is particularly poor (as discussed further in Appendix D.3).[8] When the loss component is used to identify clusters, we find that applying higher regularization to the initial ERM model further improves clustering quality, as this regularization "pushes examples further apart" along the loss direction, and adopt this convention in our experiments.[9]

In each experiment, we select the representation and the number of clusters k based on the parameter setting that achieves the highest average per-cluster Silhouette score. (For all experiments, we set the number of UMAP neighbors to 10 and the minimum distance to 0; further information about these hyperparameters can be found in [33].)

The fact that simply using the "loss component" can yield reasonable results is arguably surprising, as this essentially amounts to just picking the examples that the original network got wrong (or closer to wrong than others). Nevertheless, especially on tasks with severe data imbalances and "spurious features" (e.g., Waterbirds and CelebA), the rare subclasses do tend to be misclassified at far higher rates, so simply picking the misclassified examples can be a crude but effective heuristic.

2. *Global Clustering*: For each superclass, we search over $k \in 2, \ldots, 10$ to find the clustering that yields the highest average Silhouette score, using the dimensionality reduction procedure identified above. We similarly perform a search over clustering techniques (k-means, GMM, etc.), and find that GMM models achieve high average Silhouette scores most often in our applications. Given that GMM clustering also aligns with our theoretical analysis, we use this approach for all datasets. We refer to this global clustering as $f_{C,G}$.

3. *Overclustering*: For each superclass, we take the clustering $f_{C,G}$ achieving the highest average Silhouette score, and then split each cluster c_i into F sub-clusters $c_{i1}, \ldots, c_{iF}$, where F denotes the "overclustering factor" (fixed to 5 for all experiments). For each sub-cluster c_{ij} whose Silhouette score exceeds the Silhouette score of the corresponding points in the original clustering, and which contains at least s_{min} points (for a small threshold value s_{min}), the global clustering $f_{C,G}$ is updated to include c_{ij} as a new cluster (and its points are removed from the base cluster c_i). The overclustering factor F was coarsely tuned via visual inspection of clustering outputs (without referencing the true subclass labels); the threshold value s_{min} is used to prevent extremely small clusters, as these can lead to instability when training with GDRO and/or highly variable estimates of validation cluster-robust accuracy. (Note: We do not apply overclustering to 1-dimensional representations, as it tends to create strange within-interval splits.)

B.3.5 BiT Details

As an alternative to representations from a trained ERM model, we explore the use of BiT embeddings [27], as discussed in Section 6.5. We use the ResNet-50 version of BiT embeddings; specifically, BiT embeddings are the activations of the penultimate layer of a network pretrained on massive quantities of image data (see [27] for more details). The remainder of GEORGE proceeds the same as usual: the embeddings are clustered and then the cluster assignments are used in the GDRO objective.

For BiT, we experimented with both clustering the BiT embeddings directly (under the hypothesis that the BiT embedding space itself is a good representation), and clustering after dimensionality reduction with UMAP. We found clustering raw embeddings generally performed somewhat better; thus, we show results for clustering the raw embeddings. Due to the high dimensionality of these embeddings (2048-d), we use k-means clustering when clustering the BiT embeddings, although the rest of our procedure remains the same.

We find that BiT embeddings significantly improve the end-to-end robust performance results on CelebA; however, they perform worse than the standard version of GEORGE on all other datasets, indicating that the task-specific information is important for these other tasks to learn a "good" representation that can be clustered to find superclasses. Indeed, we find that on these other tasks, the BiT clustering is worse than

[8] We experimented with concatenating the UMAP and loss representations, but found this to reduce performance.

[9] The loss component is used for Waterbirds and CelebA (non-BiT version). For both datasets, the weight decay and learning rate used for the high-regularization ERM model are the same as the ones used for the GDRO models on that dataset.

clustering the activations of the ERM model, in terms of precision and recall at identifying poorly-performing subclasses. [For example, when BiT embeddings are used on MNIST, the "8"s are never identified as their own cluster.]

Surprisingly, the clustered BiT embeddings uniformly have a much lower Silhouette score than the clustered ERM embeddings, even for CelebA. Thus, our current unsupervised representation and clustering selection technique would not have identified the BiT embeddings as better for CelebA. Improving the representation and clustering selection metric to do a better job at automatically choosing among different representations is an interesting avenue for future work. We note that if a small validation set with *subclass* labels is available, such a set could be used to select between different clusterings by measuring the degree of overlap of the clusters with the true subclasses, as well as used to measure which representation and clustering technique eventually leads to the best validation robust accuracy; however, in general we do not assume any prior knowledge about the subclasses in this work.

B.3.6 GDRO Training Details

In the final step of GEORGE, we train a new model (with the same architecture) using the group DRO approach of [43] with weak subclass labels provided by our cluster assignments, and compare to GDRO models trained using (a) superclass labels only, (b) random subclass labels. and (c) human-annotated subclass labels. Below, we describe the hyperparameter search procedure for each such model and dataset. Unless otherwise stated, all other hyperparameters (batch size, # epochs, etc.) are the same as those for ERM.

U-MNIST In the case of U-MNIST, we ran a hyperparameter search over weight decay in [1e-3, 1e-4, 1e-5], and C (the group size adjustment parameter from [43]) in [0, 1, 2]. We find performance to be fairly insensitive to the hyperparameters, so choose weight decay of 1e-5 and $C = 0$ for simplicity and consistency with ERM.

Waterbirds For Waterbirds, we use hyperparameters provided by [43], so no additional hyperparameter tuning is required. These hyperparameters are presented in Table 5.

CelebA For CelebA, we again use hyperparameters provided by [43], so no additional tuning is required. These are presented in Table 5.

ISIC Each type of ISIC model is hyperparameter searched over the same space as the original ERM model, in addition to searching over group size adjustment parameter C in [0, 1, 2]. We found performance to be fairly insensitive to both. Hyperparameters with highest validation performance were used in the final runs, and are reported in Table 5.

B.4 Hyperparameters

In Table 5, we present the selected hyperparameters for the final runs of each dataset and method.

C Additional Experimental Results

In this section, we provide additional ablation experiments.

Dataset	Training Procedure	Epochs	Learning Rate	Batch Size	Weight Decay	Group Adj. Parameter
U-MNIST	ERM	100	2e-3	128	1e-5	-
U-MNIST	Random-GDRO	100	2e-3	128	1e-5	0
U-MNIST	Superclass-GDRO	100	2e-3	128	1e-5	0
U-MNIST	GEORGE	100	2e-3	128	1e-5	0
U-MNIST	Subclass-GDRO	100	2e-3	128	1e-5	0
Waterbirds	ERM	300	1e-3	128	1e-4	-
Waterbirds	Random-GDRO	300	1e-5	128	1	2
Waterbirds	Superclass-GDRO	300	1e-5	128	1	2
Waterbirds	GEORGE	300	1e-5	128	1	2
Waterbirds	Subclass-GDRO	300	1e-5	128	1	2
ISIC	ERM	20	1e-3	16	1e-3	-
ISIC	Random-GDRO	20	1e-3	16	1e-3	2
ISIC	Superclass-GDRO	20	1e-3	16	1e-3	1
ISIC	GEORGE	20	5e-4	16	1e-3	1
ISIC	Subclass-GDRO	20	5e-4	16	1e-3	2
CelebA	ERM	50	1e-4	128	1e-4	-
CelebA	Random-GDRO	50	1e-5	128	0.1	3
CelebA	Superclass-GDRO	50	1e-5	128	0.1	3
CelebA	GEORGE	50	1e-5	128	0.1	3
CelebA	Subclass-GDRO	50	1e-5	128	0.1	3

Table 5: Final hyperparameters used in experiments. (Note that for each dataset, all GEORGE runs use the same hyperparameters regardless of whether they use BiT or ERM embeddings.)

C.1 GEORGE results and details

C.1.1 U-MNIST

Dimensionality reduction for this dataset used 2 UMAP components and no loss component, as UMAP achieved higher SIL scores. Our clustering procedure consistently identifies a cluster with a high proportion of the low-frequency "8" subclass. As detailed in the main body, we also often observe a small additional cluster with a high concentration of "7"s written with crosses through the main vertical bar (see Figure 6); performance on this subset is low (below 90%), which explains why cluster-robust performance actually underestimates the true subclass performance on U-MNIST.

C.1.2 Waterbirds

Dimensionality reduction for this dataset used only 1 component (the loss component); this significantly outperformed UMAP both in terms of SIL score and final robust performance. We observe that while our procedure does not yield clusters with absolutely high frequencies of the minority classes (as shown in Table 2), GEORGE still identifies clusters with high enough precision (i.e., high enough proportions of the poorly-performing subclasses) such that the second stage of GEORGE can substantially improve performance on these subclasses.

C.1.3 ISIC

Dimensionality reduction for this dataset used 2 UMAP components and no loss component. On ISIC, although clustering reveals the non-patch and histopathology subclasses with fairly high fidelity as shown in Table 2, we do not observe significant improvements in performance on either subset. We hypothesize that this is due to these subsets being "inherently harder." For example, we find that even Subclass-GDRO, which uses the true patch vs. non-patch subclass labels, fails to significantly improve performance on the non-patch subclass compared to ERM, and in fact fails to significantly reduce the training loss on it compared to ERM despite being explicitly trained to do so. This suggests that the issue causing underperformance on these subsets may be due to other factors than the training optimization algorithm (such as model capacity).

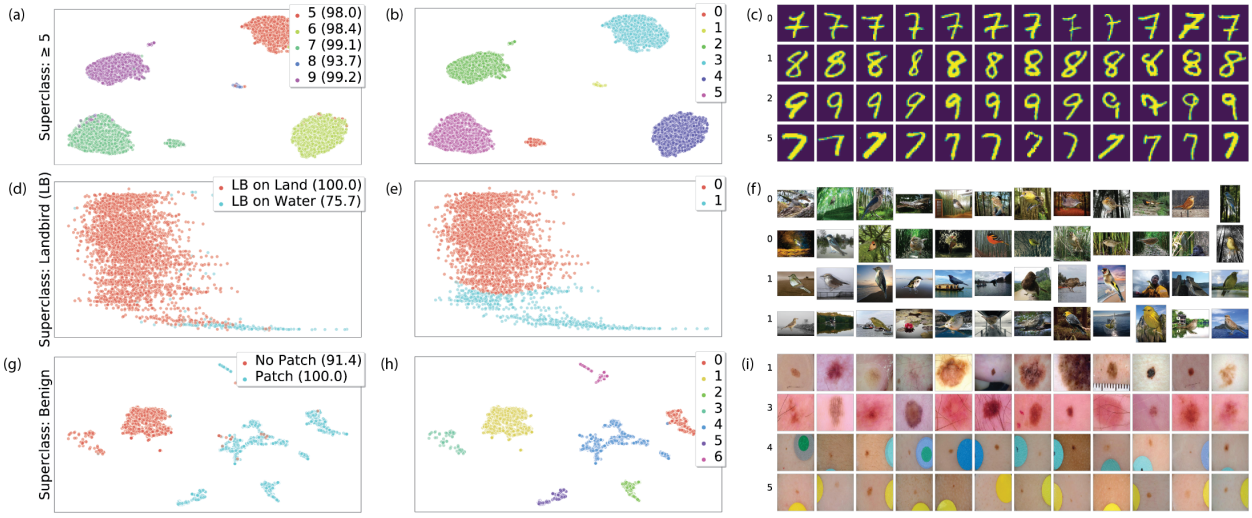


Figure 6: True subclasses in the "feature space" of a trained ERM model. Left panel legend colors points by their true subclass, and displays the validation accuracies that the model attains on each subclass. Middle panel colors points by the cluster index that GEORGE assigns them. Right panel displays randomly selected examples from each cluster. Datasets: U-MNIST (row 1), Waterbirds (row 2), and ISIC (row 3). Note that for Waterbirds, the vertical axis is the "loss component" and the horizontal axis is the UMAP component.

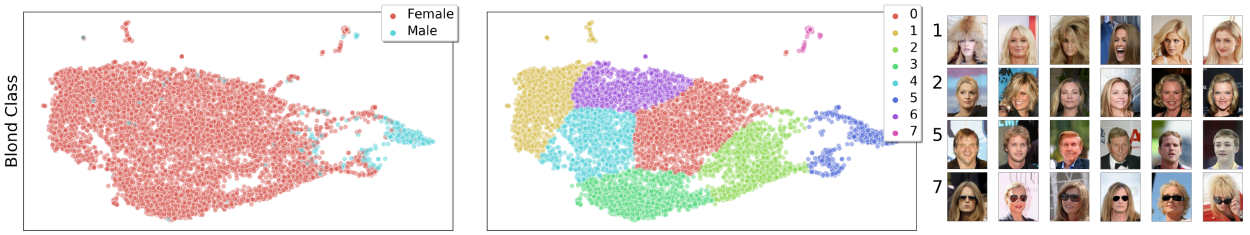


Figure 7: True subclasses in **BiT** embedding space for CelebA, clusters (middle), and examples from selected clusters (right).

C.1.4 CelebA

Dimensionality reduction for CelebA (without BiT) used only 1 component (the loss component). We observe that clustering does not do a good job of identifying the subclasses of either superclass; thus, it is not surprising that the default version GEORGE (i.e., without BiT) performs poorly. In fact, GEORGE performs poorly even compared to the non-ERM baselines. By contrast, GEORGE-BiT does significantly better; the clustering on the (nearly balanced) non-blond superclass attains approximately 95% accuracy at distinguishing between men and women, and the clustering on the blond superclass also significantly improves over the default version of GEORGE.

C.2 Additional Evaluations and Ablations

C.2.1 Visualizing Clusters

In Figures 6 and 7, we visualize the representations returned by GEORGE, as well as the clusters it finds and representative examples from each cluster.

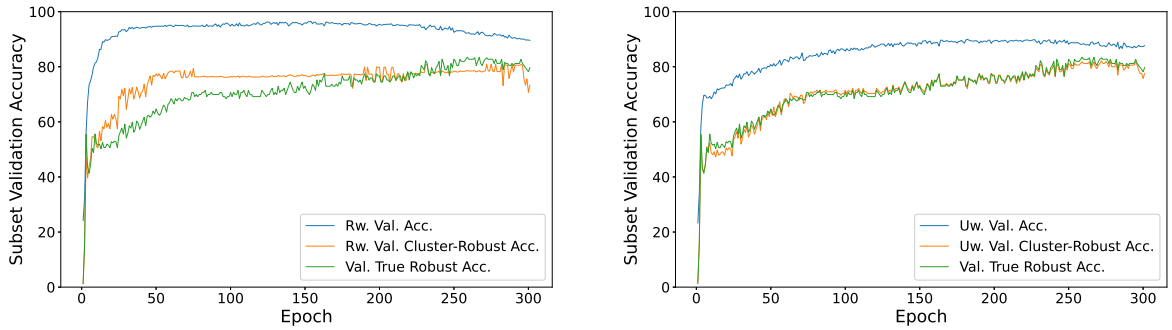


Figure 8: Overall accuracy, worst-case cluster accuracy, and worst-case true subclass accuracy on the validation set during a randomly selected training run of GEORGE on Waterbirds [(a) using reweighting for validation set, and (b) not using reweighting]. The worst-case cluster accuracy closely tracks the worst-case true subclass accuracy (especially when reweighting is not applied, whereas the overall accuracy is significantly higher due to the overrepresentation of the "easier" subclasses.

C.2.2 Comparing Cluster-Robust Performance and True Robust Performance

In addition to the results of Table 3 which show that the cluster-robust performance is a good approximation for the true robust performance, we find that the cluster-robust performance typically tracks closely with the true robust performance *throughout training* (with the exception of CelebA without BiT clusters). For example, Figure 8 plots the validation cluster-robust accuracy and validation true robust accuracy from a randomly selected training run on Waterbirds. Both metrics are quite close to each other throughout training (while the overall accuracy is significantly higher).

C.2.3 Runtime

In Table 6, we present runtimes for the standard version of GEORGE broken down by stage.

As the default implementation of GEORGE involves first training an ERM model, dimensionality-reducing and then clustering its activations, and then training a "robust" model with GDRO, the total runtime is roughly 2-3x long as that of simply training an ERM model. For GEORGE-BiT, no ERM model is trained (and we do not apply dimensionality reduction), so the runtime is just the runtime of the clustering stage plus the runtime of training the GDRO model ("Step 2"). On our datasets, the total runtime of GEORGE-BiT is less than 1.5x times the runtime of GEORGE. For instance, on CelebA with BiT embeddings (the largest and most expensive dataset), the entire clustering stage (including the time taken to compute the BiT embeddings of the datapoints) takes 46 minutes, while the time taken to train the ERM model is roughly 2.5 hours. (Clustering the BiT embeddings is more expensive because they are 2048-dimensional.)

Note that the runtime of typical clustering algorithms scales superlinearly in the number of datapoints; while the clustering runtime is usually less than the training time for the datasets we evaluate on, a remedy for larger datasets could be to only use a random subset of the data for clustering (which typically does not significantly worsen the cluster quality). In addition, we did not attempt to optimize the dimensionality reduction and clustering routines themselves. As we search over k from 2 to 10 for each superclass, and then overcluster, this is 20 different clusterings in total, along with computing the Silhouette score for each one (which is also expensive as it involves computing pairwise distances). If we instead fixed k (for instance), the total clustering runtime would be less than 7 minutes even for CelebA with BiT embeddings.

The runtime of GEORGE can be substantially reduced by training the second (robust) model for fewer epochs. On Waterbirds and CelebA, we can recover over 70% of the worst-case performance improvement of GEORGE even when we limit the total runtime to 1.3x that of ERM, simply by training for fewer epochs in the second stage. On U-MNIST, if we additionally adjust the LR decay schedule so that decay occurs before the end of the shortened training, and fix k to 5 to avoid the expensive search over k as described above, we can achieve this as well. (On ISIC, the ERM model itself already attains nearly the same robust AUROC on

the histopathology subclass, and higher on the non-patch subclass, than the GEORGE model.)

Dataset	ERM total runtime	Clustering total runtime	GDRO total runtime
U-MNIST	9m	6m	10m
ISIC	31m	2m	32m
Waterbirds	90m	1m	91m
CelebA	177m	17m	179m

Table 6: Average runtimes for different stages of GEORGE (standard implementation, without BiT). All runtimes are reported on a machine with 8 CPUs and a single NVIDIA V100 GPU, except for CelebA which was run on a machine with 32 CPUs and four NVIDIA V100 GPUs. (GEORGE-BiT differs only in the clustering runtime.)

C.2.4 Label Noise

We ran experiments in which a fixed percentage of the data of each subclass was randomly given an incorrect superclass label. With a minor modification (discarding small clusters), GEORGE empirically works well in the presence of label noise when the total number of corrupted labels in each superclass is less than the size of the smallest subclass. Up to this noise threshold, GEORGE attains +3 points robust accuracy on MNIST and +4 points robust AUROC on ISIC compared to ERM. However, ensuring subgroup-level robustness if there is a larger group of "wrong" examples is difficult because differentiating "real" subclasses from noise becomes challenging. Thus, we do not consider applying label noise to Waterbirds as the smallest subclass (waterbirds on land) is only 1% of the data; similarly, the smallest subclass on CelebA (blond males) is only 3% of the data.

In fact, our clustering approach can even be used to help identify incorrectly labeled training examples. First, if a small "gold" set of correctly labeled examples is available, the clustering found on the training data could be evaluated on this gold set; clusters consisting of mostly incorrectly labeled training examples should have very few members in the gold set. If such a "gold" set is not available, the clusters still allow for much more rapid inspection of the data for incorrect labels, since a few representative examples from each cluster can be inspected instead of a brute-force search through all the training images for incorrectly labeled images. Finally, if one has prior knowledge of the frequency of the rarest subclass in the training data, one can simply discard training examples belonging to poorly-performing clusters smaller than this threshold, treating them as incorrectly labeled.

C.2.5 Fixing k

If the number of clusters k is held fixed (rather than automatically chosen based on Silhouette score), robust performance tends to initially improve with k, before decreasing as large values of k cause fragmented clusters that are less meaningful. For example, robust accuracies on U-MNIST using 2, 5, 10, 25, and 100 clusters per superclass are 95.0%, 96.3%, 95.9%, 94.4%, 90.8% respectively. We also observe similar trends on the other datasets.

C.2.6 Effect of Model Choice on Subclass Recovery

As suggested in Section 4, choosing an appropriate model class $\mathcal{F}$ for the featurizer f_θ is important. In particular, $\mathcal{F}$ should ideally contain the inverse of the true generative function g, in order to recover the latent features $\vec{V}$ from the data X. We demonstrate the importance of model architecture on the ability to separate subclasses in the model feature space by comparing the feature representations of two simple networks on a superclass of the U-MNIST dataset (described in Section 6.1). Figure 9 shows that the choice of model family can strongly affect the learned feature representation of the initial model and its ability to provide useful information about the subclass. On this dataset, the feature space of a simple fully connected network (Figure 9a) yields substantially less separation between the known subclasses than does that of a simple convolutional network (Figure 9b), which displays clusters that clearly correspond to semantically meaningful subclasses.

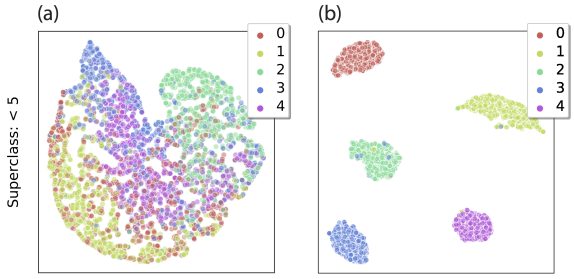


Figure 9: LeNet5 vs. LeNet 300-100 activations on U-MNIST with true subclass labels (superclass "< 5"). A simple convolutional network (LeNet5, right) separates the true subclasses well in feature space, while a network consisting only of fully-connected layers (LeNet 300-100, left) does not.

C.2.7 Additional Classification Metrics

In Table 7, we compare GEORGE and ERM in terms of per-subclass averaged accuracy (SCAA) and average precision. As expected, GEORGE slightly decreases average precision, as it trades off some average-case performance for better worst-case performance, and GEORGE typically increases per-subclass averaged accuracy (except on U-MNIST, where there is a very slight decrease), due to the fact that it significantly improves performance on poorly-performing subclasses while only slightly decreasing performance on other subclasses.

Method	Metric	Waterbirds	U-MNIST	CelebA (BiT)
GEORGE	SCAA	86.7	97.9	90.6
	AP	.967	.998	.835
ERM	SCAA	83.8	98.2	80.5
	AP	.984	.999	.912

Table 7: Per-subclass averaged accuracy (SCAA) is the mean of the accuracies on each subclass. AP denotes the average precision score (which has a maximum of 1).

C.2.8 Empirical Validation of Lemma 1

Recall that Lemma 1 says that if we know the true data distribution, we can estimate the per-subclass loss R_c by the quantity $\tilde{R}_c$, a reweighted average of the losses in superclass $S(c)$ based on the ratio of the posterior likelihood of a point given subclass c to its posterior likelihood given superclass $S(c)$; Lemma 1 bounds their difference in terms of the number of datapoints n. In Figure 10, we empirically validate Lemma 1 on a synthetic mixture-of-Gaussian example (in which the true data distribution is indeed known). We generate data in dimension $d = 3$ with two subclasses each containing six subclasses each, and compute R_c and $\tilde{R}_c$ for varying numbers of samples n in order to observe the scaling with n. We average results over 20 trials; in each trial, new per-subclass distributions are randomly sampled and then new datapoints are sampled. Results are shown in Figure 10. As can be observed from the log-log plot, the slope of the line corresponding

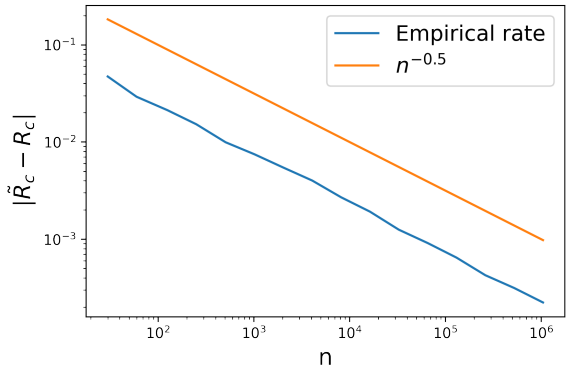


Figure 10: Comparison of theoretical and simulated convergence of $\tilde{R}_c - R_c$.

to the simulated $|\tilde{R}_c - R_c|$ value is very close to that of the predicted rate; the best fit line has a coefficient of -0.5065, corresponding to a $O(n^{-0.5065})$ rate, essentially matching Lemma 1's predicted $O(n^{-0.5})$ rate.

D Derivations and Proofs

D.1 Analysis of Example 3.1.

We restate Example 3.1 from Section 3.2 below:

Example 3.1 The binary attribute vector $\vec{Z}$ has dimension 2, i.e., $\vec{Z} = (Z_1, Z_2)$, while only Z_2 determines the superclass label Y, i.e., $Y = Z_2$. The latent attribute Z_1 induces two subclasses in each superclass, each distributed as a different Gaussian in feature space, with mixture proportions α and $1 - \alpha$ respectively. For linear models with regularized logistic loss, as the proportion α of the rare subclasses goes to 0, the worst-case subclass accuracy of ERM is only $O(\alpha)$, while that of GDRO is $1 - O(\alpha)$.

Proof. Specifically, we consider the following distribution setup: $\vec{Z} \in \{-1, +1\}^2$, with $P(\vec{Z} = (-1, -1)) = P(\vec{Z} = (+1, +1)) = \frac{1-\alpha}{2}$, $P(\vec{Z} = (-1, +1)) = P(\vec{Z} = (+1, -1)) = \alpha/2$, and $P(V_1|Z_1) = \mathcal{N}(4Z_1, \alpha^2)$, $P(V_2|Z_1, Z_2) = \mathcal{N}(Z_1 + 3Z_2, \alpha^2)$, and the label $Y = h(Z_1, Z_2)$ simply equals Z_2. We assume the observed data $X = (V_1, V_2)$, i.e., the observed data is the same as the "underlying features" $\vec{V}$.

Thus, the superclass $Y = -1$ is made up of a "big" subclass with distribution $\mathcal{N}((-4, -4), \alpha^2\mathbf{I})$ and relative mixture weight $1 - \alpha$ [corresponding to $\vec{Z} = (-1, -1)$], and a "small" subclass with distribution $\mathcal{N}((+4, -2), \alpha^2\mathbf{I})$ and relative mixture weight α [corresponding to $\vec{Z} = (+1, -1)$], where $\mathbf{I}$ denotes the 2×2 identity matrix. The superclass $Y = +1$ is made up of a "big" subclass with distribution $\mathcal{N}((+4, +4), \alpha^2\mathbf{I})$ and relative mixture weight $1 - \alpha$ [corresponding to $\vec{Z} = (+1, +1)$], and a "small" subclass with distribution $\mathcal{N}((-4, +2), \alpha^2\mathbf{I})$ and relative mixture weight α [corresponding to $\vec{Z} = (-1, +1)$].

For notational simplicity in the following analysis, we will henceforth rename the label $Y = -1$ as $Y = 0$. The prediction of the logistic regression model on a given sample (x_1, x_2) is $\sigma(w_1 x_1 + w_2 x_2) = \sigma(w^T x)$, where $\sigma(x) := \log(\frac{1}{1+e^{-x}})$ denotes the sigmoid function and w_1, w_2 are the weights of the model. The decision boundary is the line $w^T x = 0$; examples with $w^T x < 0$ are classified as $Y = 0$, else they are classified as $Y = 1$. [For simplicity of exposition, we assume there is no bias term, and assume that we regularize the norm of the classifier so that $\|w\|_2 \leq R$ for some constant R, as changing the parameter norm does not change the decision boundary. Note that neither assumption is necessary, but they serve to simplify the analysis.]

The logistic loss is the negative log-likelihood, which is $-\sum_i \big(y_i \log(\frac{1}{1+e^{-w^T x}}) + (1 - y_i) \log(\frac{1}{1-e^{-w^T x}})\big) =$

$\sum_i \big(y_i \log(1+e^{-w^T x}) + (1-y_i)\log(1-e^{-w^T x})\big)$. Notice that by symmetry, the loss on the two subclasses with $Z_1 = Z_2$ is the same, as is the loss on the two subclasses with $Z_1 \neq Z_2$. Therefore, we focus on the class $Y = 1$. The expected average loss on the $Y = 1$ superclass is $\mathbb{E}_{x|y=1}[\log(1+e^{-w^T x})] = P(Z_1 = 1|Z_2 = 1) \cdot \mathbb{E}_{x|(z_1,z_2)=(1,1)}[\log(1+e^{-w^T x})] + P(Z_1 = -1|Z_2 = 1) \cdot \mathbb{E}_{x|(z_1,z_2)=(-1,1)}[\log(1+e^{-w^T x})]$.

By 1-Lipschitz continuity of the logistic loss and Jensen's inequality,

$$\left|\mathbb{E}_{x|y=1,z=1}[\log(1+e^{-w^T x})] - \mathbb{E}_{x|y=1,z=1}[\log(1+e^{-w^T(4,4)})]\right| \leq$$
$$\mathbb{E}_{x|y=1,z=1}[|w^T x - w^T(4,4)|] =$$
$$\mathbb{E}_{x|y=1,z=1}[|w_1(x_1-4)| + |w_2(x_2-4)|] = (|w_1| + |w_2|)\mathbb{E}[|b|],$$

where $b \sim N(0, \alpha^2)$. $\mathbb{E}[|b|] = \alpha\sqrt{2/\pi}$; so, the loss on the $Z_1 = 1$ subclass is bounded in the range $\log(1+e^{-w^T(4,4)}) \pm \alpha\sqrt{2/\pi} \cdot \|w\|_2$.

Similarly, the loss on the $Z_1 = -1$ subclass is bounded in the range $\log(1+e^{-w^T(-4,2)}) \pm \alpha\sqrt{2/\pi} \cdot \|w\|_2$. So, the total loss is bounded in $(1-\alpha)\log(1+e^{-w^T(4,4)}) + \alpha\log(1+e^{-w^T(-4,2)}) \pm \alpha\sqrt{2/\pi} \cdot \|w\|_2$. When α is sufficiently small, the first term is $\Theta(1)$, while the latter two are $O(\alpha)$ (under the assumption that $\|w\|_2$ is bounded). For a fixed value of $\|w\|_2$, the first term is minimized when $w/\|w\|_2 = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$, so that $w^T(4,4)$ is as large as possible. A $\Theta(\alpha)$-scale perturbation to the direction $w/\|w\|_2$ results in an increase of $\Theta(\alpha)$ to the term $(1-\alpha)\log(1+e^{-w^T(4,4)})$. Thus, whenever α is sufficiently small, $w/\|w\|_2$ must be $(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}) + O(\alpha)$ in order to minimize the loss subject to the $\|w\|_2 \leq R$ constraint. In other words, the regularized ERM solution converges to a multiple of $(w_1, w_2) = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}})$ as $\alpha \downarrow 0$.

For the $Z_1 = -1$ subclass, $w^T x$ is a normal random variable with mean $-4w_1 + 2w_2$ and variance $\alpha\|w\|_2^2$. When α is sufficiently small and $w/\|w\|_2 = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}) + O(\alpha)$, the quantity $-4w_1 + 2w_2$ is negative with magnitude $O(1)$—and thus, since examples with $w^T x < 0$ are classified as $Y = 0$, this means that for sufficiently small α the fraction of the subclass $Z_1 = -1$ classified correctly as $Y = 1$ is only $O(\alpha)$.

By contrast, the GDRO solution minimizes the maximum per-subclass loss. Since each subclass has the same covariance $\alpha^2\mathbf{I}$, the GDRO decision boundary is the line that separates the superclass means and has maximum distance to any subclass mean. After normalization to have $\|w\|_2 = 1$, this is the line $(-\frac{1}{\sqrt{5}}, \frac{4}{\sqrt{5}})$; the true solution will be some multiple of this (depending on α and R), giving rise to the same boundary. As $\alpha \downarrow 0$, the accuracy of this decision boundary is $1 - O(\alpha)$, since the variance of each subclass is $O(\alpha^2\mathbf{I})$. □

D.2 Proofs from Section 5

D.2.1 Proof of Lemma 1

Lemma 1. *Let R_c be the sample average loss of examples in subclass c. Let $w(x,c) := \frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))}$. Let $\tilde{R}_c$ be the sample average of $w(x_i,c)\ell(f(x_i),y_i)$ over all examples x_i with* superclass *label $y_i = S(c)$. Then $\tilde{R}_c$ is an unbiased estimate of R_c, and their difference converges to 0 at $O(1/\sqrt{n})$.*

Proof. Define $\#_{y=k} := \sum_{i=1}^n \mathbf{1}(y_i = k)$ and $\#_{z=c} := \sum_{i=1}^n \mathbf{1}(z_i = c)$. Using this notation, $R_c = \frac{1}{\#_{z=c}} \sum_{i=1, z_i=c}^n \ell(f(x_i), S(c))$, and $\tilde{R}_c = \frac{1}{\#_{y=S(c)}} \sum_{i=1, y_i=S(c)}^n \frac{\mathcal{P}(x_i|z_i=c)}{\mathcal{P}(x_i|y_i=S(c))}\ell(f(x_i), S(c))$. First, observe that $\mathbb{E}[\tilde{R}_c] = \mathbb{E}[R_c]$: the expectation of each term in the summation defining $\mathbb{E}[\tilde{R}_c]$ is $\mathbb{E}_{x\sim[\mathcal{P}|y=S(c)]}\left[\frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))}\ell(f(x), S(c))\right] =$

$\int_{\mathbb{R}^d} \frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))} \ell(f(x), S(c)) \mathcal{P}(x|y=S(c))\, dx = \int_{\mathbb{R}^d} \mathcal{P}(x|z=c)\ell(f(x), S(c))\, dx = \mathbb{E}_{x\sim[\mathcal{P}(x|z=c)]}[\ell(f(x), S(c))] = \mathbb{E}[R_c]$, and so $\mathbb{E}[\tilde{R}_c] = \mathbb{E}[R_c]$.

Now, note that $\frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))} \geq \mathcal{P}(z=c|y=S(c)) \geq \pi$. Thus, assuming ℓ is bounded, each term in the summation defining $\tilde{R}_c$ is bounded, and has mean $\mathbb{E}[R_c]$ as argued above; applying Hoeffding's inequality (to bound the probability that a sum of bounded random variables deviates from its mean by more than a specified amount) yields that $\left|\tilde{R}_c - \mathbb{E}[R_c]\right| \leq O(\frac{1}{\sqrt{n}})$ with high probability. Similarly, applying Hoeffding's inequality to R_c yields that $|R_c - \mathbb{E}[R_c]| \leq O(\frac{1}{\sqrt{n}})$ with high probability. □

D.2.2 Proof of Theorem 1

We restate Theorem 1 below:

Theorem 1. *Let $\hat{R}_{robust} := \max_c \hat{R}_c$. Suppose ℓ, f are Lipschitz, f has bounded parameters, and $\mathcal{P}(x|z=c)$ is Gaussian and unique for each subclass c. Then, if we estimate $\hat{\mathcal{P}}$ using the algorithm from [3], $\hat{f} := \min_{f\in\mathcal{F}} \hat{R}_{robust}(f)$ satisfies $R_{robust}(\hat{f}) - \min_{f\in\mathcal{F}} R_{robust}(f) \leq \tilde{O}(\sqrt{1/n})$ w.h.p.*

Recall that we define $\hat{R}_c$ to be the same as $\tilde{R}_c$ except with weights $\hat{w}(x,c)$ computed from $\hat{\mathcal{P}}$. More precisely, $\hat{R}_c := \frac{1}{\#_{y=S(c)}} \sum_{i=1, y_i=S(c)}^{n} \hat{w}(x,c)\ell(f(x_i), S(c))$ where $\hat{w}(x,c) := \frac{\hat{\mathcal{P}}(x|z=c)}{\hat{\mathcal{P}}(x|y=S(c))}$. [In Appendix D.5, we provide an efficient algorithm to minimize $\hat{R}_{\text{robust}} = \max_c \hat{R}_c$.]

Our strategy to prove Theorem 1 will be as follows. First, we prove a general statement (that holds regardless of the form of the true data distribution) that relates the total variation (TV) between the true per-subclass distributions and the estimated per-subclass distributions to the difference between $\tilde{R}_c$ and the *perturbed* loss $\hat{R}_c$ (Lemma 2). Next, we bound the total variation between the true and estimated per-subclass distributions in the mixture-of-Gaussians case, using the Gaussian mixture learning algorithm from [3]. Finally, we use standard uniform convergence-type results to yield the final high probability bound on the robust risk of the returned model $\hat{f}$.

D.2.2.1 Total variation estimation error to error in loss

Lemma 2. *Let $\pi_{\min}$ be the minimum true subclass proportion and $\hat{\pi}_{\min}$ be the minimum estimated subclass proportion. Suppose that we have estimated subclass-conditional distributions $\hat{\mathcal{P}}(x|z)$ such that for each subclass c, there exists c' such that $\text{TV}(\mathcal{P}(x|z=c), \hat{\mathcal{P}}(x|z=c')) \leq \epsilon$. If ℓ is globally bounded by M, then for fixed f, $|\hat{R}_{c'} - \tilde{R}_c| \leq \frac{2MC}{\min\{\pi_{\min}, \hat{\pi}_{\min}\}} \cdot \epsilon + O(\frac{1}{\sqrt{n}})$ with high probability.*

Proof. First, we bound $|\hat{w}(x,c') - w(x,c)| = \left|\frac{\hat{\mathcal{P}}(x|z=c')}{\hat{\mathcal{P}}(x|y=S(c))} - \frac{\mathcal{P}(x|z=c)}{\mathcal{P}(x|y=S(c))}\right| = \frac{1}{\mathcal{P}(x|y=S(c))}\left|\frac{\mathcal{P}(x|y=S(c))}{\hat{\mathcal{P}}(x|y=S(c))} \cdot \hat{\mathcal{P}}(x|z=c') - \mathcal{P}(x|z=c)\right|$. By triangle inequality, this is bounded above by $\frac{1}{\mathcal{P}(x|y=S(c))}\Big(\left|\frac{\mathcal{P}(x|y=S(c))}{\hat{\mathcal{P}}(x|y=S(c))} \cdot \hat{\mathcal{P}}(x|z=c') - \hat{\mathcal{P}}(x|z=c')\right| + |\hat{\mathcal{P}}(x|z=c') - \mathcal{P}(x|z=c)|\Big)$.

Note that by definition $\hat{\mathcal{P}}(x|y=b) = \sum_{i\in S_b} \hat{\pi}_{b,i}\hat{\mathcal{P}}(x|z=c_i)$, where we define $\hat{\pi}_{b,i} := \hat{\mathcal{P}}(z=c_i|y=b)$ for each $i \in S_b$. In words, $\hat{\pi}_{b,i}$ is the "mixture weight" of subclass i within the distribution of superclass b. We can thus bound $\left|\frac{\mathcal{P}(x|y=S(c))}{\hat{\mathcal{P}}(x|y=S(c))} - 1\right|\hat{\mathcal{P}}(x|z=c')$ by $\frac{1}{\hat{\mathcal{P}}(z=c|y=S(c))}\left|\frac{\mathcal{P}(x|y=S(c))}{\hat{\mathcal{P}}(x|y=S(c))} - 1\right|\hat{\mathcal{P}}(x|y=S(c)) = \frac{1}{\hat{\mathcal{P}}(z=c|y=S(c))}\left|\hat{\mathcal{P}}(x|y=S(c)) - \mathcal{P}(x|y=S(c))\right| \leq \frac{1}{\hat{\pi}_{\min}}\left|\hat{\mathcal{P}}(x|y=S(c)) - \mathcal{P}(x|y=S(c))\right|$.

So,

$$\begin{aligned}\mathbb{E}[|\hat{w}(x,c') - w(x,c)|] &\leq \frac{1}{\hat{\pi}_{\min}}\mathbb{E}\left[\frac{1}{\mathcal{P}(x|y=S(c))} \cdot (|\hat{\mathcal{P}}(x|y=S(c)) - \mathcal{P}(x|y=S(c))| + |\hat{\mathcal{P}}(x|z=c') - \mathcal{P}(x|z=c)|)\right] \\ &= \frac{1}{\hat{\pi}_{\min}}\int_{\mathbb{R}^d} \frac{1}{\mathcal{P}(x|y=S(c))} \cdot \left(|\hat{\mathcal{P}}(x|y=S(c)) - \mathcal{P}(x|y=S(c))| + |\hat{\mathcal{P}}(x|z=c') - \mathcal{P}(x|z=c)|\right) \cdot \mathcal{P}(x|y=S(c))\, dx \\ &\leq \frac{C}{\hat{\pi}_{\min}}\epsilon,\end{aligned}$$

since $\int_{\mathbb{R}^d} \left|\hat{\mathcal{P}}(x|z=c) - \mathcal{P}(x|z=c)\right| dx = 2TV(\hat{P}(x|z=c), \hat{P}(x|z=c')) \leq 2\epsilon$ by assumption, and $|\hat{\mathcal{P}}(x|y=S(c)) - \mathcal{P}(x|y=S(c))|$ is the weighted sum of the total variations between the distributions of each subclass of $S(c)$, of which there are $\leq C-1$ [assuming there are at least two superclasses, since otherwise the "classification problem" is trivial].

Now, $|\hat{R}_{c'} - R_c| \leq \frac{1}{\#_{y=S(c)}} \sum_{i|y_i=S(c)} |\hat{w}(x_i, c_i') - w(x_i, c_i)|\ell(f(x_i), S(c))$. The expectation of each term in the summation is therefore bounded above by $\frac{2MC}{\hat{\pi}_{\min}}\epsilon$, since the loss is globally bounded by M. Finally, applying Hoeffding's inequality yields that $|\hat{R}_{c'} - \tilde{R}_c| \leq \frac{2MC}{\hat{\pi}_{\min}}\epsilon + O(\frac{1}{\sqrt{n}})$ with high probability. □

Total variation in estimated per-subclass distributions: Gaussian case

[3] provides an algorithm for estimating mixtures of Gaussians; they show that $\tilde{O}(kd^2/\epsilon^2)$ samples are sufficient to learn a mixture of k d-dimensional Gaussians to within error ϵ in total variation. Concretely, given n samples from a mixture-of-Gaussian distribution $\mathcal{P}$ and the true number of mixture components k, the algorithm in [3] returns a k-component mixture-of-Gaussian distribution $\hat{\mathcal{P}}$ such that $TV(\mathcal{P}, \hat{\mathcal{P}}) \leq \tilde{O}(\sqrt{1/n})$. To prove Theorem 1, we use this result and bound the overall total variation error in terms of the maximum per-component total variation error. The proof depends on the key lemma stated below (whose proof appears at the end of this section). We then apply Lemma 2 to translate this total variation error bound to a bound on the robust loss of the end model.

In order to apply Lemma 2, we first need to relate the total variation error ϵ between the mixtures to the total variation error between the individual mixture components; we show that when ϵ is small enough, then the total variation error between the mixture components is $O(\epsilon)$ as well. We state this formally in Lemma 3 (proved later in this section).

Lemma 3. *Let $\mathcal{P}$ and $\hat{\mathcal{P}}$ be two k-component Gaussian mixtures, and suppose the k components of $\mathcal{P}$, denoted by $p_1, \ldots, p_k$, are* distinct *Gaussian distributions and all have nonzero mixture weights. Similarly denote the k components of $\hat{\mathcal{P}}$ by $\hat{p}_1, \ldots, \hat{p}_k$. There exists a constant $c(\mathcal{P})$ depending only on the parameters of $\mathcal{P}$ such that for all sufficiently small $\epsilon > 0$, whenever $TV(\mathcal{P}, \hat{\mathcal{P}}) \leq \epsilon$ it is the case that $\max_{1\leq i\leq k} \min_{1\leq j\leq k} TV(p_i, \hat{p}_j) \leq c(\mathcal{P}) \cdot \epsilon$.*

In addition, we use the following standard result from learning theory [29] to relate the minimizer of the estimated robust training loss $\hat{R}_{\text{robust}}$ to the minimizer of the true robust training loss R_{robust}.

Lemma 4. *Suppose $g(\cdot,\cdot) \in [-B, B]$. Let $f(\theta) := \mathbb{E}_{(x,y)\sim P}[g(x, y; \theta)]$ and let $\hat{f}(\theta) := \frac{1}{n}\sum_{i=1}^{n} g(x_i, y_i; \theta)$, where $\{(x_i, y_i)\}_{i=1}^n$ are IID samples from P. Suppose $g(\cdot,\cdot\theta)$ is L-Lipschitz w.r.t. θ, so $f(\theta), \hat{f}(\theta)$ are L-Lipschitz. Then with probability $\geq 1 - O(e^{-p})$, we have*

$$\forall \theta \text{ s.t. } \|\theta\| \leq R, \quad |\hat{f}(\theta) - f(\theta)| \leq O\left(B\sqrt{\frac{p\log(nLR)}{n}}\right) \tag{5}$$

Theorem 1 Proof

Using the preceding lemmas, we will now prove Theorem 1.

Proof. First, we estimate $\hat{\mathcal{P}}$ using the Gaussian mixture learning algorithm from [3]. With high probability, this returns a k-component mixture-of-Gaussian distribution $\hat{\mathcal{P}}$ such that $\mathrm{TV}(\mathcal{P}, \hat{\mathcal{P}}) \leq \tilde{O}(\sqrt{1/n})$. By Lemma 3, if n is large enough then this means that for each subclass c there exists a subclass c' such that $\mathrm{TV}(\mathcal{P}(x|z=c), \hat{P}(x|z=c')) \leq \tilde{O}(1/\sqrt{n})$. So, applying Lemma 2, we have that for a fixed function f, $|\hat{R}_c - \tilde{R}_c| \leq \tilde{O}(1/\sqrt{n})$, for all subclasses c (WLOG we may reorder the components of $\hat{P}$ so that $c = c'$ for all c, for notational convenience). By Lemma 1 and triangle inequality, $|\hat{R}_c - R_c|$ is $\tilde{O}(1/\sqrt{n})$ as well. Thus, for any given f, $|\hat{R}_{\mathrm{robust}}(f) - R_{\mathrm{robust}}(f)| = |\max_c \hat{R}_c - \max_c R_c| = \tilde{O}(1/\sqrt{n})$ with high probability.

Let $R^*_{\mathrm{robust}}(f)$ denote the true *population* robust loss. Then Lemma 4 says that $|R_{\mathrm{robust}}(f) - R^*_{\mathrm{robust}}(f)|$ with high probability for all f in the hypothesis class $\mathcal{F}$. Similarly, we can use an analogous uniform convergence result to show that $|\hat{R}_{\mathrm{robust}}(f) - R_{\mathrm{robust}}(f)| \leq \tilde{O}(1/\sqrt{n})$ for all f in the hypothesis class with high probability. Thus, by triangle inequality and union bound, $|\hat{R}_{\mathrm{robust}}(f) - R^*_{\mathrm{robust}}(f)| \leq \tilde{O}(1/\sqrt{n})$ for all f in the hypothesis class with high probability, and in particular this holds for the minimizer $\hat{f}$ of $\hat{R}_{\mathrm{robust}}$.

Thus, under the given assumptions, the excess robust generalization risk (i.e., worst-case subclass generalization risk) of the GEORGE model $\hat{f}$ is $\tilde{O}(1/\sqrt{n})$ [which is near-optimal in terms of sample complexity, since $\Omega(1/\sqrt{n})$ is a generic worst-case lower bound even if the subclass labels are known].

Note that a technical requirement of the above argument is that the samples we use to estimate $\hat{\mathcal{P}}$ should be independent from those we use to compute the robust loss; for this to hold, we may randomly sample half of the examples to learn the distribution $\hat{\mathcal{P}}$ (and its mixture components), and then use the other half to minimize the robust loss. This does not change the asymptotic dependency on the number of samples n. [In practice, however, we use all examples in both phases, to get the most out of the data.] □

Lemma 3 Proof

Before we prove Lemma 3, we first provide a simple lemma bounding the total variation distance of two Gaussians in terms of the Euclidean distance between their parameters, directly based on the results from [13].

Lemma 5. *Let p be a d-dimensional Gaussian with mean μ and full-rank covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$. Let p' be another Gaussian with mean μ' and covariance Σ'. Then there exists a constant $c(\mu, \Sigma)$ [i.e., depending only on the parameters of p] such that for all sufficiently small $\epsilon > 0$, whenever $\|\mu - \mu'\|_2 \leq \epsilon$ and $\|\Sigma - \Sigma'\|_F \leq \epsilon$ it is the case that $TV(p, p') \leq c(\mu, \Sigma) \cdot \epsilon$.*

Proof. The one-dimensional case is shown in Theorem 1.3 of [13]. The higher-dimensional case follows from Theorems 1.1 and 1.2 of [13]. Note that the constant c does not depend on ϵ, although it may depend on d. □

For convenience, we restate Lemma 3 below.

Lemma 3. *Let $\mathcal{P}$ and $\hat{\mathcal{P}}$ be two k-component Gaussian mixtures, and suppose the k components of $\mathcal{P}$, denoted by $p_1, \ldots, p_k$, are* distinct *Gaussian distributions and all have nonzero mixture weights. Similarly denote the k components of $\hat{\mathcal{P}}$ by $\hat{p}_1, \ldots, \hat{p}_k$. There exists a constant $c(\mathcal{P})$ depending only on the parameters of $\mathcal{P}$ such that for all sufficiently small $\epsilon > 0$, whenever $TV(\mathcal{P}, \hat{\mathcal{P}}) \leq \epsilon$ it is the case that $\max_{1\leq i\leq k} \min_{1\leq j\leq k} TV(p_i, \hat{p}_j) \leq c(\mathcal{P}) \cdot \epsilon$.*

Proof. The case $k = 1$ is vacuous (since the "mixture" is simply a single Gaussian); so, suppose $k > 1$. Denote the mixture weights of the true distribution $\mathcal{P}$ as $m_1, \ldots, m_k$, and the mean and covariance parameters of the individual distributions in $\mathcal{P}$ as $\mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k$. In other words, for $x \in \mathbb{R}^d$, $\mathcal{P}(x) = \sum_{i=1}^{k} m_i \mathcal{N}_{\mu_i,\Sigma_i}(x)$, where $m_i \in (0, 1)$ and $\mathcal{N}_{\mu,\Sigma}$ denotes the normal density with mean μ and covariance Σ. Similarly, denote the mixture weights of the estimated distribution $\hat{\mathcal{P}}$ by $\hat{m}_1, \ldots, \hat{m}_k$, and the mean and covariance parameters of $\hat{\mathcal{P}}$ by $\hat{\mu}_1, \ldots, \hat{\mu}_k, \hat{\Sigma}_1, \ldots, \hat{\Sigma}_k$. For simplicity assume the covariance matrices Σ_i of each component in the true distribution are strictly positive definite (although this is not required).

Define $q(m'_1, \dots, m'_k, \mu'_1, \dots, \mu'_k, \Sigma'_1, \dots, \Sigma'_k) = \int_{\mathbb{R}^d} \left| \sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \Sigma_i}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \Sigma'_i}(x) \right| dx$. The domain of q is constrained to have $m'_i \in [0, 1]$ for all $1 \le i \le k$ and $\sum_{i=1}^k m'_i = 1$, as well as to have Σ'_i be SPD. By definition, $q(\hat{m}_1, \dots, \hat{m}_k, \hat{\mu}_1, \dots, \hat{\mu}_k, \hat{\Sigma}_1, \dots, \hat{\Sigma}_k)$ is simply twice the total variation between $\mathcal{P}$ and $\hat{\mathcal{P}}$.

Note that, since we assumed the mixture components are unique and $m_i \neq 0$ for all i, the only global minima of q [where q evaluates to 0, which means that $\mathcal{P}$ and $\hat{\mathcal{P}}$ are the same distribution] are where $(m'_{\pi(i)}, \mu'_{\pi(i)}, \Sigma'_{\pi(i)}) = (m_i, \mu_i, \Sigma_i)$ for all $1 \le i \le k$, for some permutation π—in other words, when the two distributions have the exact same mixture components and mixture weights up to permutation. Further, it is not hard to see that the ϵ-sublevel sets of q are contained within compact sets when ϵ is sufficiently small, and therefore $\lim_{\epsilon \to 0} \{(m'_1, \dots, \mu'_1, \dots, \Sigma'_1, \dots) : q(m'_1, \dots, \mu'_1, \dots, \Sigma'_1, \dots) \le \epsilon\}$ is exactly the set of global minima of q. Finally, note that q is continuous on its domain. Thus, for a fixed distribution $\mathcal{P}$, as $\epsilon \to 0$, the set of points such that $q(m'_1, \dots, \mu'_1, \dots, \Sigma'_1, \dots) \le \epsilon$ is the union of sets of radius $O(\delta(\epsilon))$ around each of the global minima of q, where $\delta(\epsilon) \to 0$ as $\epsilon \to 0$. In other words, the set of all Gaussian mixtures $\mathcal{P}'$ with $TV(\mathcal{P}, \mathcal{P}') \le \epsilon$ is the set of all mixtures $\mathcal{P}'$ whose parameters $\{m'_i, \mu'_i, \Sigma'_i\}$ are $O(\delta(\epsilon))$-close to those of the true distribution $\mathcal{P}$, up to permutation. In particular, if $TV(\mathcal{P}, \mathcal{P}') \le \epsilon$, then for each individual Gaussian component $\mathcal{N}_{\mu_i, \Sigma_i}$ in $\mathcal{P}$, there exists a component $\mathcal{N}_{\mu'_j, \Sigma'_j}$ in $\mathcal{P}'$ whose parameters are $O(\delta(\epsilon))$-close to it, i.e., $|m_i - m_j| + \left\|\mu_i - \mu'_j\right\|_2 + \left\|\Sigma'_i - \Sigma'_j\right\|_F \le O(\delta(\epsilon))$.

We now argue that $\lim_{\epsilon \to 0} \frac{\delta(\epsilon)}{\epsilon}$ must be a constant (i.e., δ is $\Theta(\epsilon)$ as $\epsilon \to 0$) in order for the total variation between the two mixtures to be $\le \epsilon$. We do so by Taylor expanding a set of quantities whose magnitudes lower bound the total variation between $\mathcal{P}$ and $\mathcal{P}'$, and showing that these quantities are locally linear in the parameter differences between $\mathcal{P}'$ and $\mathcal{P}$ when these differences are sufficiently small.

By assumption, $2TV(\mathcal{P}, \mathcal{P}') = q(m'_1, \dots, m'_k, \mu'_1, \dots, \mu'_k, \Sigma'_1, \dots, \Sigma'_k) = \int_{\mathbb{R}^d} \left| \sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \sigma_i^2}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x) \right| dx$ $\le 2\epsilon$. Notice that $2TV(\mathcal{P}, \mathcal{P}') = \sup_{S \subseteq \mathfrak{M}^d} \int_S \left| \sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \sigma_i^2}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x) \right| dx \ge$ $\sup_{S \subseteq \mathfrak{M}^d} \left| \int_S \left(\sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \sigma_i^2}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x) \right) dx \right|$, where $\mathfrak{M}^d$ denotes the collection of all measurable subsets of $\mathbb{R}^d$.

Suppose for now that $d = 1$. Then, $\sup_{S \subseteq \mathfrak{M}} \left| \int_S \left(\sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \sigma_i^2}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x) \right) dx \right| \ge$ $\sup_{c_j \in \mathbb{R}} \left| \int_{-\infty}^{c_j} \left(\sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \sigma_i^2}(x) - \sum_{i=1}^k m'_i \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x) \right) dx \right| = \sup_{c_j \in \mathbb{R}} |h(\mathcal{P}', c_j)|$, where we define $h(\mathcal{P}'; c_j) =$ $h(m'_1, \dots, m'_k, \mu'_1, \dots, \mu'_k, \sigma_1^{2\prime}, \dots, \sigma_k^{2\prime}; c_j) := \sum_{i=1}^k m_i \int_{-\infty}^{c_j} \mathcal{N}_{\mu_i, \sigma_i^2}(x)\, dx - \sum_{i=1}^k m'_i \int_{-\infty}^{c_j} \mathcal{N}_{\mu'_i, \sigma_i^{2\prime}}(x)\, dx$. So, $|h(\mathcal{P}'; c_j)|$ is a lower bound on twice the total variation between $\mathcal{P}$ and $\mathcal{P}'$, for any value of c_j. Let $\vec{v} \in \mathbb{R}^{3k}$ denote the vector of parameters $(m_1, \dots, m_k, \mu_1, \dots, \mu_k, \sigma_1^2, \dots, \sigma_k^2)$, and similarly for $\vec{v}'$. For ease of notation we write $h(\vec{v}'; c_j)$ interchangeably with $h(\mathcal{P}'; c_j)$ and $h(m'_1, \dots, m'_k, \mu'_1, \dots, \mu'_k, \sigma_1^{2\prime}, \dots, \sigma_k^{2\prime}; c_j)$.

Assume $TV(\mathcal{P}, \mathcal{P}') \le \epsilon$, so as argued before $\|\vec{v}' - \pi(\vec{v})\|_2$ is $O(\delta(\epsilon))$ for some permutation π and some function δ with $\lim_{x \to 0} \delta(x) = 0$. [More precisely, $\lim_{\epsilon \to 0} \max_{\vec{v}' : TV(\mathcal{P}, \mathcal{P}') \le \epsilon} \min_{\pi : \pi(k)} \|\vec{v}' - \pi(\vec{v})\|_2 = 0$.] Without loss of generality, we will henceforth simply write $\vec{v}$ in place of $\pi(\vec{v})$. For notational simplicity, let's write $\delta := \|\vec{v}' - \vec{v}\|_2$. As h is smooth around $\vec{v}$, we can Taylor expand h about the point $\vec{v} = \vec{v}'$ [i.e., $\{m'_1 = m_1, ..., \mu'_1 = \mu_1, ..., \sigma_1^{2\prime} =$

$\sigma_1^2, ..., \sigma_k^{2\prime} = \sigma_k^2\}]$ to get

$$h(m_1', \ldots, m_k', \mu_1', \ldots, \mu_k', \sigma_1^{2\prime}, \ldots, \sigma_k^{2\prime}; c_j) = h(\vec{v}') = h(\vec{v}; c_j) + \nabla h(\vec{v}; c_j)^T(\vec{v}' - \vec{v}) + O(\|\vec{v}' - \vec{v}\|_2^2) \ .$$

Note that $h(\vec{v}; c_j) = 0$ (since $\vec{v}$ are the true parameters).

$\nabla h(\vec{v}; c_j)^T = (\frac{\partial}{\partial m_1'} h(\vec{v}; c_j), \ldots, \frac{\partial}{\partial \mu_1'} h(\vec{v}; c_j), \ldots, \frac{\partial}{\partial \sigma_1'} h(\vec{v}; c_j), \ldots)$.

$\frac{\partial}{\partial m_i'} h(\vec{v}; c_j) = \int_{-\infty}^{c_j} p_{\mu_i, \sigma_i^2}(x)\, dx = \frac{1}{2}\text{erfc}\left(\frac{\mu_i - c_j}{\sqrt{2}\sigma_i}\right)$. [$\sigma_i$ denotes the positive root of σ_i^2.]

$\frac{\partial}{\partial \mu_i'} h(\vec{v}; c_j) = -\frac{m_i e^{-(c_j - \mu_i)^2/(2\sigma_i^2)}}{\sqrt{2\pi}\sigma_i}$.

$\frac{\partial}{\partial \sigma_i^{2\prime}} h(\vec{v}; c_j) = \frac{m_i e^{-(c_j - \mu_i)^2/(2\sigma_i^2)}(\mu_i - c_j)}{2\sqrt{2\pi}\sigma_i^3}$.

Define $f_i(x) = \frac{1}{2}\text{erfc}\left(\frac{\mu_i - x}{\sqrt{2}\sigma_i}\right)$ for $1 \le i \le k$, $-\frac{m_i e^{-(x-\mu_i)^2/(2\sigma_i^2)}}{\sqrt{2\pi}\sigma_i}$ for $k+1 \le i \le 2k$, and $\frac{m_i e^{-(x-\mu_i)^2/(2\sigma_i^2)}(\mu_i - x)}{2\sqrt{2\pi}\sigma_i^3}$ for $2k+1 \le i \le 3k$. So $\nabla h(\vec{v}; c_j)^T = (f_1(c_j), \ldots, f_{3k}(c_j))^T$.

We have $|\nabla h(\vec{v}; c_j)^T \delta + O(\|\delta\|_2^2)| \le 2\epsilon$ for all $c_j \in \mathbb{R}$. Now, we claim that it is possible to select $c_1, \ldots, c_{3k} \in \mathbb{R}$ such that the $3k \times 3k$ matrix with rows $\nabla h(\vec{v}; c_j)^T$ for $1 \le j \le 3k$ is nonsingular.

Proof: Suppose that there is a nonzero vector $\vec{w} \in \mathbb{R}^{3k}$ such that $(f_1(x), \ldots, f_{3k}(x))^T w = 0$ for all $x \in \mathbb{R}$. This means that the function $w_1 f_1(x) + \cdots + w_{3k} f_{3k}(x)$ is identically 0. But this is impossible unless $\vec{w} = 0$, as the f_i's form a linearly independent set of functions (which can easily be seen by looking at their asymptotic behavior). Thus, there is no nonzero vector that is orthogonal to every vector in the set $\bigcup_{x \in \mathbb{R}} \{(f_1(x), \ldots, f_{3k}(x))\}$. In particular, this means that we can find $c_1, \ldots, c_{3k} \in \mathbb{R}$ such that the matrix with rows $(f_1(c_j), \ldots, f_{3k}(c_j))$ is nonsingular (i.e., has linearly independent rows).

Call this matrix A. So $\|A\delta + \eta\|_\infty \le 2\epsilon = 2\epsilon \|\mathbb{1}\|_\infty$ where $\eta \in \mathbb{R}^{3k}$ is such that $\|\eta\|_\infty$ is $O(\|\delta\|_2^2) = O(\|\delta\|_\infty^2)$, and $\mathbb{1}$ denotes the vector of all ones in $\mathbb{R}^{3k}$. So $\|\delta\|_\infty - \left\|A^{-1}\eta\right\|_\infty \le \left\|\delta + A^{-1}\eta\right\|_\infty \le 2\epsilon \left\|A^{-1}\right\|_\infty \|\mathbb{1}\|_\infty = 2\epsilon \left\|A^{-1}\right\|_\infty$, where $\left\|A^{-1}\right\|_\infty$ is the induced ∞-norm of A^{-1}, and the first inequality follows from the reverse triangle inequality. Note that $\left\|A^{-1}\eta\right\|_\infty \le \left\|A^{-1}\right\|_\infty \|\eta\|_\infty \le O(\|\delta\|_\infty^2)$ since A^{-1} is defined independently of δ.

So, $\|\delta\|_\infty - O(\|\delta\|_\infty^2) \le 2\epsilon \left\|A^{-1}\right\|_\infty$, and thus $\|\delta\|_\infty$ [which is, by definition, the maximum error in any parameter $m_1, \ldots, m_k, \mu_1, \ldots, \mu_k, \sigma_1^2, \ldots, \sigma_k^2$ up to permutation] is $O(\epsilon)$. But then, the total variation between each pair of mixture components $\mathcal{N}_{\mu_i, \sigma_i^2}$ and $\mathcal{N}_{\mu_i', \sigma_i^{2\prime}}$ is also $O(\epsilon)$, by Lemma 5.

Thus, when $d = 1$, if the total variation between the two Gaussian mixtures $\mathcal{P}$ and $\hat{\mathcal{P}}$ is $O(\epsilon)$, the total variation between each mixture component must also be $O(\epsilon)$ [where the big-O notation suppresses all parameters that depend on the true distribution $\mathcal{P}$], as desired. (Note that total variation is always between 0 and 1.)

Now suppose $d > 1$. Similarly to before, we have $2\epsilon \ge 2TV(\mathcal{P}, \mathcal{P}') \ge \max_{S \subseteq \mathfrak{M}^d} \left| \int_S \left(\sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \Sigma_i}(x) - \sum_{i=1}^k m_i' \mathcal{N}_{\mu_i', \Sigma_i'}(x) \right) dx \right|$

$\ge \max_{\vec{c}_j \in \mathbb{R}^d} \left| \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} \left(\sum_{i=1}^k m_i \mathcal{N}_{\mu_i, \Sigma_i}(x) - \sum_{i=1}^k m_i' \mathcal{N}_{\mu_i', \Sigma_i'}(x) \right) dx \right| = \max_{\vec{c}_j \in \mathbb{R}^d} |h(\mathcal{P}', \vec{c}_j)|$, where we define $h(\mathcal{P}'; \vec{c}_j)$ as $\sum_{i=1}^k m_i \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} \mathcal{N}_{\mu_i, \Sigma_i}(x)\, dx - \sum_{i=1}^k m_i' \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} \mathcal{N}_{\mu_i', \Sigma_i'}(x)\, dx$. Again, we equivalently denote this by $h(\vec{v}'; c_j)$, where $\vec{v}' \in \mathbb{R}^{k + kd + kd(d+1)/2} := (m_1', ..., \text{vec}(\mu_1'), ..., \text{vec}(\Sigma_1'), ...)$ denotes the parameters collected into a single vector. As before, we Taylor expand about the point $\vec{v}' = \vec{v}$ to get that $|\nabla h(\vec{v}; \vec{c}_j)^T \delta + O(\|\delta\|_\infty^2)| \le 2\epsilon$, where $\delta := \vec{v}' - \vec{v}$.

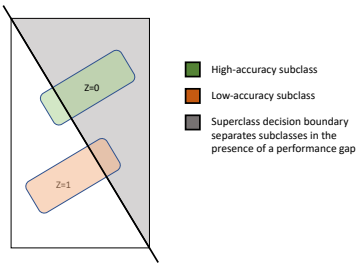


Figure 11: A performance gap between subclasses within the same superclass implies a corresponding degree of separation in feature space. Green and red are true subclasses for the superclass which the model predicts as the gray region; the decision boundary for the superclass classification task also approximately separates the subclasses.

Now, we compute the entries of $\nabla h(\vec{v}; \vec{c}_j)$:
$\frac{\partial}{\partial m_i'} h(\vec{v}; \vec{c}_j) = \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} \mathcal{N}_{\mu_i, \Sigma_i}(x)\, dx = \frac{1}{(2\pi)^{d/2}\sqrt{|\det(\Sigma_i)|}} \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} e^{-(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)/2}\, dx.$
$\frac{\partial}{\partial (\mu_i')_a} h(\vec{v}; \vec{c}_j) = \frac{m_i}{(2\pi)^{d/2}\sqrt{|\det(\Sigma_i)|}} \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} (\Sigma_i^{-1})_a \cdot (x-\mu_i) e^{-(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)/2}\, dx$, where $(\mu_i)_a$ denotes the a^{th} entry of the vector μ_i and $(\Sigma_i^{-1})_a$ is the a^{th} row of the matrix Σ_i^{-1}.
$\frac{\partial}{\partial (\Sigma_i')_{ab}} h(\vec{v}; \vec{c}_j) = -\frac{m_i (\Sigma_i')^{-1}_{ab}}{2(2\pi)^{d/2}\sqrt{|\det(\Sigma_i)|}} \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} e^{-(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)/2}\, dx +$
$\frac{m_i}{(2\pi)^{d/2}\sqrt{|\det(\Sigma_i)|}} \int_{-\infty}^{c_{j1}} \int_{-\infty}^{c_{j2}} \cdots \int_{-\infty}^{c_{jd}} [(\Sigma_i^{-1}(x-\mu_i))(\Sigma_i^{-1}(x-\mu_i))^T]_{ab}\, e^{-(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)/2}\, dx$, where M_{ab} denotes the (a, b) entry of the matrix M (note that all matrices involved in this expression are symmetric).

Once again, this set of $k + kd + kd(d+1)/2$ partial derivatives, considered as functions of $\vec{c}_j$, comprise a linearly independent set of functions, since the (μ_i, Σ_i) pairs are unique. The remainder of the proof proceeds analogously to the $d = 1$ case. □

D.3 Subclass Performance Gaps Enable Distinguishing Between Subclasses

In this section, we give simple intuition for why a performance gap between two subclasses of a superclass implies that it is possible to discriminate between the two subclasses in feature space to a certain extent.

For example, suppose the setting is binary classification, and one of the superclasses has two subclasses with equal proportions in the dataset. Suppose we have access to a model whose training accuracy on one subclass is x, while its training accuracy on the other subclass is y, where $1 \geq x > y \geq 0$.

Of the correctly classified examples, $\frac{x}{x+y} > \frac{1}{2}$ fraction of them are from the first subclass; similarly, of the incorrectly classified examples, $\frac{1-y}{2-x-y} > \frac{1}{2}$ fraction of them are from the second subclass.

This means that if we form "proxy subclasses" by simply splitting the superclass into the correctly classified training examples and incorrectly classified training examples, the resulting groups can in fact be a good approximation of the true subclasses! This is illustrated in Figure 11. For instance, suppose $x = 0.9$ and $y = 0.6$. Then $\frac{x}{x+y} = 0.6$ and $\frac{1-y}{2-x-y} = 0.8$ - so, 60% of the examples in the first group are from subclass 1, and 80% of those in the second group are from subclass 2, which is much better than randomly guessing the true subclasses (in which the concentration of each subclass in each guessed group will approach 50% as $n \to \infty$). In the extreme case, if one subclass has accuracy 1 and the other has accuracy 0, then the superclass decision boundary separates them perfectly (no matter their proportions).

Combined with other information, this helps explain why looking at the way each example is classified (such as the loss of the example or related error metrics) can be helpful to discriminate between the subclasses.

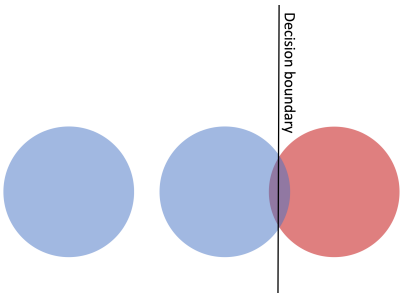


Figure 12: "Inherent hardness": the red and blue superclasses overlap, making it impossible to distinguish between them with perfect overall accuracy. The blue superclass has two subclasses; on the leftmost subclass, the classifier can attain perfect accuracy.

D.4 Inherent Hardness

We define the "inherent hardness" of a (task, function class) pair as the minimum attainable robust error, i.e.,

$$\operatorname*{argmax}_{f \in \mathcal{F}} \min_{c \in \{1,\dots,C\}} \mathbb{E}_{(x,y)|z=c} \left[\mathbf{1}(f(x) = y) \right],$$

where the function class is denoted by $\mathcal{F}$. (This can be thought of as the "Bayes robust risk.") We allow the function f to be stochastic: i.e., for a given input x, it may output a fixed probability distribution over the possible labels, in which case we define $\mathbf{1}(f(x) = y)$ as the probability assigned by f to the label y, given input x. By definition, the inherent hardness lower bounds the robust error attained by any classifier in $\mathcal{F}$, regardless of how it is trained or how much data is available. The only way to improve robust performance is therefore to either make the model class $\mathcal{F}$ more expressive (i.e., include more functions in $\mathcal{F}$) or to collect new data such that the covariates x include more information that can be used to distinguish between different classes. (Of course, both of these changes would be expected to improve overall performance as well, if sufficient data is available.) Thus, addressing hidden stratification effects caused by "inherent hardness" is beyond the scope of this work. A simple example of an "inherently hard" task (i.e., a task with nonzero "inherent hardness") is shown in Figure 12; no classifier can get perfect accuracy on every subclass, because the two superclasses overlap and thus it is impossible to distinguish between them in the region of overlap. Nevertheless, it is possible to attain perfect accuracy on *some* subclasses in this example, meaning that there will still be performance gaps between the subclasses.

D.5 GDRO with soft group assignments

As shown above in Appendix D.2, we can minimize $\max_{c \in [C]} \mathbb{E}_{x \sim \hat{P}_{S(c)}}[\hat{w}(x,c)\ell(x, S(c); \theta)]$ as a surrogate for $\max_{c \in [C]} \mathbb{E}_{x \sim P_c}[\ell(x, S(c); \theta)]$. Here, $\hat{\mathcal{P}}_{S(c)}$ is the empirical distribution of training examples of superclass c, $\hat{w}(x,c)$ is shorthand for $\dfrac{\hat{\mathcal{P}}(x|z=c)}{\hat{\mathcal{P}}(x|y=S(c))}$, and $\ell(x, S(c); \theta)$ is shorthand for $\ell(f_\theta(x), S(c))$, where f_θ is a classifier parameterized by θ. If we define the density $A_c(x,z) = \hat{\mathcal{P}}_{S(c)}(x)\mathbf{1}(z=c)$, then $\max_{c \in [C]} \mathbb{E}_{x \sim \hat{\mathcal{P}}_{S(c)}}[\hat{w}(x,c)\ell(x, S(c); \theta)] = \max_{c \in [C]} \mathbb{E}_{(x,z) \sim A_c}[\hat{w}(x,z)\ell(x, S(z); \theta)]$.

If we now define $\tilde{\ell}(x, z; \theta) := \hat{w}(x,z)\ell(x, S(z); \theta)$, we see that this falls directly within the group DRO framework of [43]. We thus obtain Algorithm 2, which is a minor modification of Algorithm 1 of [43].

The weights $\hat{w}(x,c)$ correspond to "soft labels" indicating the probability a particular example came from a particular superclass; notice that that $\mathbb{E}_{(x,z) \sim A_c}[\hat{w}(x,z)\ell(x, S(z); \theta)]$ depends on every training example in the *superclass* $S(c)$, so each training example is used in multiple terms in the maximization.

Finally, note that if the assumptions (informally: nonnegativity, convexity, Lipschitz continuity, and boundedness) of Proposition 2 in [43] hold for the modified loss $\tilde{l}(x, z; \theta)$, then the convergence guarantees carry over

Algorithm 2 Modified Group DRO

Input: Step sizes η_q, η_θ; empirical per-superclass distributions $\hat{P}_b$ for each superclass $b \in [B]$
Initialize $\theta^{(0)}$ and $q^{(0)}$
for $t = 1, \ldots, T$ **do**
  $c \sim \text{Uniform}(1, \ldots, c)$
  $x \sim \hat{P}_{S(c)}$
  $q' \leftarrow q^{(t-1)}$
  $q'_c \leftarrow q'_c \cdot \exp\left(\eta_q \cdot \hat{w}(x, c) \cdot \ell(x, S(c); \theta^{(t-1)})\right)$
  $q^{(t)} \leftarrow q' / \sum_c q'_c$
  $\theta^{(t)} \leftarrow \theta^{(t-1)} - \eta_\theta \cdot q_c^{(t)} \cdot \hat{w}(x, c) \cdot \nabla_\theta \ell(x, S(c); \theta^{(t-1)})$
end for

as well, since Algorithm 2 is a specific instantiation of Algorithm 1 from [43]. So, under these assumptions, the convergence rate of Algorithm 2 is $O(1/\sqrt{T})$, where T is the number of iterations. [Specifically, the average iterate after T iterations achieves a robust loss that is $O(1/\sqrt{T})$ greater than that of the minimizer of the robust loss.]

In our experiments, we found hard clustering to work better than soft clustering; as it also has the advantage of simplicity, all evaluations were performed with hard clustering.