Adaptive Risk Minimization: A Meta-Learning Approach for Tackling Group Shift

Marvin Zhang*[1], **Henrik Marklund***[2], **Abhishek Gupta**[1], **Sergey Levine**[1], **Chelsea Finn**[2]
[1] UC Berkeley, [2] Stanford University

Abstract

A fundamental assumption of most machine learning algorithms is that the training and test data are drawn from the same underlying distribution. However, this assumption is violated in almost all practical applications: machine learning systems are regularly tested on data that are structurally different from the training set, either due to temporal correlations, particular end users, or other factors. In this work, we consider the setting where test examples are not drawn from the training distribution. Prior work has approached this problem by attempting to be robust to all possible test time distributions, which may degrade average performance, or by "peeking" at the test examples during training, which is not always feasible. In contrast, we propose to learn models that are *adaptable*, such that they can adapt to distribution shift at test time using a batch of unlabeled test data points. We acquire such models by learning to adapt to training batches sampled according to different sub-distributions, which simulate structural distribution shifts that may occur at test time. We introduce the problem of adaptive risk minimization (ARM), a formalization of this setting that lends itself to meta-learning methods. Compared to a variety of methods under the paradigms of empirical risk minimization and robust optimization, our approach provides substantial empirical gains on image classification problems in the presence of distribution shift.

Adaptable models that can handle distribution shift at test time time

1 Introduction

The standard assumption in empirical risk minimization (ERM) is that the data distribution at test time will match their distribution at training time. When this assumption does not hold, the performance of standard ERM methods typically deteriorates rapidly, e.g., [26, 27], and this setting is commonly referred to as distribution or dataset *shift* [51, 35]. For instance, we can imagine an image classification system that, after training on a large database of past images, is deployed to specific end users. Each user takes photos with differing cameras, locations, and subjects, leading to shift in the input distribution. This test scenario must be carefully considered when building machine learning systems for real world applications.

* Standard assumption in ML models

Algorithms for handling distribution shift have been studied under a number of frameworks [51]. Many of these frameworks, such as domain adaptation [13], assume access to unlabeled test data at training time, which are often not readily available and can be quite challenging to obtain. One prominent family of methods that avoids this assumption is distributionally robust optimization (DRO) [21, 5, 15]. DRO methods learn *robust* models by optimizing against adversarially chosen training distributions, thus these models have maximal worst case performance. However, these methods can often be overly pessimistic and learn models that do not perform well on the actual test distributions [31]. This issue of pessimism can be partially mitigated by carefully choosing the set of adversarial distributions to be robust against, motivating a number of approaches that only allow the adversary to shift the underlying group distribution, e.g., changing the distribution of attributes

} ⇒ DRO

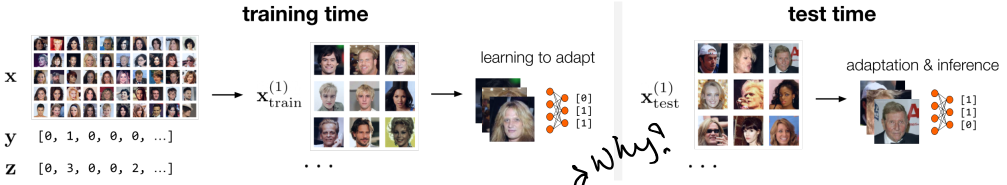
} ⇒ Cons of DRO

⇒ solution

*equal contribution

Q1. Why DRO can be pessimistic? Does this assumption holds for other adverserial examples too?

Preprint. Under review.



why?

Figure 1: A schematic of our problem setting and approach, described in detail in Section 4. Left: During training, we assume access to labeled data along with group information z, which allows us to construct training sub-distributions that exhibit group distribution shift. We show only one sub-distribution for brevity, which depicts an overemphasis on blond-haired males compared to the empirical distribution. This is analogous to the structural DRO problem setup, but unlike DRO, we use these sub-distributions to learn a model that is adaptable to distribution shift via a form of meta-learning. Right: We perform unsupervised adaptation to different test distributions, without requiring test data at training time or specializing to a single test distribution as in domain adaptation. If the test shifts we observe are similar to those simulated by the training sub-distributions, we expect that we can effectively adapt to these test distributions for better performance.

or users in the training data. This setting of *structural DRO* allows for more tractable optimization while still permitting a wide range of realistic distribution shifts [63, 31, 50, 55].

Assumptions:
(i) Availability of a batch of unlabeled test data
(ii) We can leverage ideas from meta learning

In this work, we take a different approach to combating group distribution shift by learning models that are able to deal with shift by *adapting* to the test time distribution. To do so, we assume that we can access a batch of *unlabeled* test data points *at test time* – as opposed to individual isolated inputs – which can be used to implicitly infer the test distribution. This assumption is reasonable in many standard supervised learning setups, e.g., photos from an end user are collected together into a library. As illustrated in Figure 1, we utilize ideas from meta-learning to train such a model that specializes its behavior depending on the actual test distribution, thereby suffering less from the pessimism that plagues DRO, without requiring test data at training time like domain adaptation.

Our first contribution is to formally introduce the problem of adaptive risk minimization (ARM), in which models have the opportunity to adapt to the data distribution at test time based on unlabeled data points. Second, we design a method for solving ARM that, given a set of candidate distribution shifts, meta-learns a model that is adaptable to these shifts. Our experiments demonstrate that our method is able to outperform both prior ERM and DRO methods in image classification settings exhibiting group shift at test time, including a DRO benchmark exhibiting attribute shifts [55] and a federated learning benchmark with different users at training and test time [9].

2 Related Work

Our method uses meta-learning to make it possible for models to adapt to the specific distribution of inputs seen at test time. A number of prior works have studied distributional shift in various forms [51], and in this section we review prior work in the most relevant areas, including robust optimization, domain adaptation, and meta-learning.

D ⇒ Distributionally
R ⇒ Robust
O ⇒ optimization

Robust optimization. DRO methods optimize machine learning systems to be robust to adversarial data distributions, thus optimizing for worst case performance against distribution shift [21, 5, 42, 16, 47, 15, 7]. Recent work has shown that these algorithms can be utilized with high capacity function approximators, such as neural networks, with additional care taken for regularization and model capacity [55]. Unlike DRO methods, our proposed method doesn't require the model to perform well on all test time distribution shifts, but instead trains it to adapt to these shifts.

⇒ Difference in the proposed method

Also of particular interest are methods for robustness or adaptation to different users [29, 10, 34, 48, 38, 17, 40], a setting commonly referred to as robust or fair federated learning [45, 48, 38]. Unlike these works, we consider the federated learning problem setting in which we do not assume access to any labels from any test users, as we partition users into mutually exclusive train and test sets. We argue that this is a realistic setting for many practical machine learning systems – oftentimes, the only available information from the end user is an unlabeled batch of data.

Huh! Why? ←
Bad idea IMHO

Domain adaptation. Another approach to the problem of distribution shift is to assume access to examples from the test distribution during training [51]. One prominent paradigm is domain adaptation [13], which augments the training procedure using the test examples, through approaches

Q Is adverserial setting only setting where the performance of a model is worst?

such as importance weighting the training data [61, 32, 12], representation learning [22, 3], and adversarial training [19, 67, 44]. Our method is similar to unsupervised approaches to domain adaptation, which make use of unlabeled test data. However, we do not need to specify the test distribution at training time, and we are not limited to a single test distribution, as in domain adaptation and transductive learning settings [13, 68].

Meta-learning. Meta-learning [57, 6, 66, 28] has been most extensively studied in the context of few shot supervised learning methods [56, 69, 52, 18, 70, 62, 20, 1]. These methods, in contrast to our approach, adapt using small amounts of *labeled* data. Some meta-learning methods adapt using both labeled and unlabeled data, such as [53, 71, 46, 2, 39, 4], or consider the setting where task groupings are not known, such as [23, 30, 24, 59], though these works do not focus on the same setting of distribution shift. As our work is complementary to prior meta-learning methods, we can in theory replace our meta-learner with one of these prior methods.

Our method also resembles prior meta-learning methods for domain generalization [37, 14], which studies shift at the level of test domains. Other prior methods that consider unlabeled test time adaptation typically only handle label shift [54, 41, 64] or use carefully chosen surrogate losses, relying on the correlation between this loss and test time performance [65].

3 Preliminaries

* In this paper, we focus on the supervised learning problem where, given a training dataset of N input output pairs $(x^{(i)}, y^{(i)})$ sampled i.i.d. from an unknown distribution p, the goal is to learn a model $g(\cdot\,;\theta):\ \mathcal{X} \to \mathcal{Y}$ that is parameterized by $\theta \in \Theta$ and predicts an output $y \in \mathcal{Y}$ given an input $x \in \mathcal{X}$.

Group robustness. DRO tries to learn a model that is robust to an adversarial test distribution q within some uncertainty set $\mathcal{Q}$. One popular method for specifying $\mathcal{Q}$ is through *structural DRO* or *group DRO*, which posits a categorical random variable $z \in \{1, \ldots, S\}$ and defines the adversarial distribution as $q(x, y, z) = q(z)p(x, y|z)$. This formulation is a generalization of common settings such as label shift and, arguably, most cases of covariate shift [63, 51, 58]. The fundamental assumption is that the distribution shift at test time is fully modeled by a shift in the group distribution $q(z)$ [63, 31]. $\mathcal{Q}$ may then be defined in a number of ways, but because z is discrete and finite, we often choose $\mathcal{Q}$ to simply allow for any distribution over z, i.e., any point on the $(S-1)$-dimensional probability simplex [55]. The worst case performance of g is then given by its performance for the worst realization of z. The adversarial optimization problem for structural DRO is given by

What is S?

other ways of defining uncertainity?

Assumption

Why defining z as discrete and finite is a good choice?

$$\boxed{\min_{\theta} \sup_{q \in \mathcal{Q}} \mathbb{E}_{q_z}\left[\mathbb{E}_{p_{xy|z}}[\ell(g(x;\theta), y)]\right] .}$$

Group fairness. Closely related to structural DRO are approaches that optimize for *fairness* across different groups, e.g., [72]. Methods for fairness are typically concerned with achieving more uniform accuracy across groups, which is essentially equivalent to the goal of structural DRO [38]. One prominent approach, which we refer to as *q*-fairness, specifies a hyperparameter $q \geq 0$ and constructs the optimization problem as

$$\boxed{\min_{\theta} \mathbb{E}_{p_z}\left[\mathbb{E}_{p_{xy|z}}[\ell(g(x;\theta), y)]^{q+1}\right] .}$$

$q=0 \simeq$ ERM

$q=\infty \simeq$ DRO

We can see that, if $q = 0$, then this is equivalent to ERM, and as $q \to \infty$, we recover unconstrained structural DRO. Typically, q is set to some moderate value to trade off between robustness and accuracy [38]. For brevity, in this work we will refer to structural DRO and group based q-fairness collectively as *group robustness* methods.

⇒ obvious

Meta-learning. We approach the goal of learning adaptable models through the lens of meta-learning. Supervised meta-learning considers a distribution over tasks $p(\tau)$, where each τ itself specifies a distribution over data $p(x, y|\tau)$. Given a set of training tasks τ_{train}, sampled i.i.d. from $p(\tau)$, the goal is to optimize a model such that it can quickly learn good performance on a new set of test tasks τ_{test}, also sampled i.i.d. from $p(\tau)$.[1] To evaluate quick learning, we typically only observe K data points sampled according to $p(x, y|\tau_{\text{test}})$, where K is some small number. We define a *learner* as a function $h(\cdot\,;\phi):\ \Theta \times (\mathcal{X}, \mathcal{Y})^K \to \Theta$, which is parameterized by ϕ. h takes as input the current model parameters θ and K labeled data points and produces updated parameters θ' after learning on

[1] We omit index superscripts on tasks for brevity.

1. Train a model, with θ parameters, in a supervised way
2. Learn a function h, that updates the parameters $\theta \to \theta'$ after learning on K labeled data points.

What should be the objective? of course, optimizing

3. What should be the objective : Of course, optimizing the values of θ and φ

the K points. The goal of meta-learning is to optimize both θ and ϕ such that h is able to effectively update g to achieve good performance on a new task with only a small amount of data from the task. This goal is given by the optimization problem

$$\boxed{\min_{\theta,\phi} \mathbb{E}_{p_\tau}\left[\mathbb{E}_{p_{xy|\tau}}[\ell(g(x;\theta'),y)]\right], \text{ where } \theta' = h(\theta,(x_1,y_1),\ldots,(x_K,y_K);\phi).} \qquad (1)$$

The concept of using "h" isn't new. It just differs on the implementation side

Prior meta-learning methods differ in how they implement h, with approaches such as recurrent models [56, 52], gradient based fine tuning [18, 49], and learned embeddings [69, 62]. In Section 4, we describe our setting, which borrows ideas from group robustness and meta-learning to optimize for unsupervised adaptation performance to group distribution shifts. We further present our method based on meta-learning contextual embeddings in subsection 4.2.

4 Adaptive Risk Minimization

What is z?

In our problem setting, we have access to a training dataset that consists of N labeled data points $(x^{(i)}, y^{(i)}, z^{(i)})$ sampled from the training distribution p, where, like group robustness methods, we also observe the group $z^{(i)}$ associated with each point. At test time, we are given batches of K *unlabeled* data points, where each batch is drawn from a distribution that may differ from both p and the other batch distributions, and we do not observe either y or z. For example, we can imagine a test scenario that separately considers each user's images, as discussed in Section 1.

4.1 Deriving the ARM Objective

Assumptions
1. K Test points together
2. Structural assumption

We motivate and design our training objective using several mild assumptions. Our first critical assumption is that we observe the K test points all together rather than one at a time. Second, we use the structural assumption from the group robustness setting: aligning with meta-learning terminology, we model different potential test distributions as different tasks τ_{test}, and we assume that τ_{test} specifies an unknown $p(z|\tau_{\text{test}})$. Since z is a categorical random variable, τ_{test} can be instantiated as the parameters of this categorical distribution, i.e., as a random variable S-dimensional parameter whose entries sum to 1. Thus, like group robustness, the test distribution parameterized by τ_{test} can be expressed as $p(x, y, z|\tau_{\text{test}}) = p(z|\tau_{\text{test}})p(x, y|z)$.

Finally! clears z

For brevity, let $\mathbf{x}$ denote the test batch $(x_1, \ldots, x_K)$, and define $\mathbf{y}$ and $\mathbf{z}$ analogously. Consider the distribution of *batches* of data that we observe at test time, given by the equation

$$\boxed{p_{\text{test}}(\mathbf{x}, \mathbf{y}) = \iint p(\mathbf{x}, \mathbf{y}|\mathbf{z})p(\mathbf{z}|\tau_{\text{test}})p(\tau_{\text{test}})d\mathbf{z}d\tau_{\text{test}}.}$$

We wish to train a model that can adapt using $\mathbf{x}$, drawn from this joint distribution, to better predict $\mathbf{y}$, and we must specify our training tasks accordingly. However, we do not know a priori what the test tasks will be, and we are not provided training tasks as in the standard meta-learning setting. Instead, we draw inspiration from prior work in deep learning that demonstrates that uniformly sampling over a quantity of interest, such as labels or groups, is a strong method for achieving robustness and performance with respect to that quantity [60, 8, 55]. We extend this approach to our setting by defining our training task distribution $p(\tau_{\text{train}})$ to uniformly cover many different shifts of interest.

For example, in the federated learning setting, one may reasonably assume that the only test distributions of interest are those consisting of data from only a single test user. This intuitively leads to defining $p(\tau_{\text{train}})$ to uniformly place weight on only tasks that assign probability to only a single training user. By defining this distribution over training tasks, our model is thus trained to adapt to specific individual users. In the general case, we define $p(\tau_{\text{train}})$ to be uniform over all group distributions, in order to break spurious correlations, emphasize rare groups in the training data, and achieve greater robustness.

⇒ Everyone should remember this anyways!

In practice, τ_{train} specifies a procedure for drawing weighted samples from the training data. Specifically, we define the empirical group distribution as $p_{\text{emp}}(z) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}\{z^{(i)} = z\}$, where $\mathbb{1}$ denotes the indicator function. To sample training data according to τ_{train}, we sample each training point with probability proportional to $p_{\text{emp}}(x^{(i)}, y^{(i)}, z^{(i)}|\tau_{\text{train}}) \propto \frac{p(z^{(i)}|\tau_{\text{train}})}{p_{\text{emp}}(z^{(i)})}$. → check code for this

After constructing training tasks sampled from $p(\tau_{\text{train}})$, our goal is to meta-learn ϕ and θ such that h can adapt g using *unlabeled* training data sampled according to a particular τ_{train}. Assuming that

Assumption

we will observe similar batches of data at test time, we can then perform the same unsupervised adaptation procedure for better test performance. This motivates the ARM objective, given by

$$\min_{\theta,\phi} \mathbb{E}_{p_\tau}\left[\mathbb{E}_{p_{z|\tau}}\left[\mathbb{E}_{p_{xy|z}}\left[\frac{1}{K}\sum_{k=1}^{K}\ell(g(x_k;\theta'),y_k)\right]\right]\right], \text{ where } \theta' = h(\theta, x_1, \ldots, x_K; \phi)\,. \tag{2}$$

As h does not have access to labels, we are able to evaluate the adapted model on the same K data points used for adaptation, rather than a separate task validation set. This again simulates the test time settings, where we will adapt on the same test points that we wish to predict on. Because of this distinction, we refer to h as an *adaptation model* rather than a learner. *

4.2 A Meta-Learning Approach to Optimizing the ARM Objective

Both points make sense ⟵

(1.) If h is dep on lablels, it won't be of any use at test time.

(2.) h needs to be differential w.r.t both θ and ϕ

Algorithm 1 presents a general meta-learning approach for optimizing the ARM objective. In line 5, h outputs updated parameters θ' using an unlabeled batch of data. We assume that h is a differentiable function with respect to θ and ϕ, and this allows us to meta-train both θ and ϕ for *post adaptation* performance on a mini batch of data sampled according to a particular τ_{train} (line 6). However, this adaptation is performed using unlabeled data, mimicking the test time procedure detailed in lines 7-8. In practice, we typically sample mini batches of training tasks, to provide a better gradient signal for optimizing ϕ.

Algorithm 1 Meta-Learning for Adaptive Risk Minimization

```
// Training procedure
Require: # training steps $T$, sample size $K$,
         learning rate $\eta$, training task distribution $p(\tau_{\text{train}})$
1: Initialize: $\theta, \phi$
2: for $t = 1, \ldots, T$ do
3:     Sample $\tau_{\text{train}} \sim p(\cdot)$
4:     Sample $(x_k, y_k, z_k) \sim p_{\text{emp}}(\cdot,\cdot,\cdot|\tau_{\text{train}})$ for $k = 1, \ldots, K$
5:     $\theta' \leftarrow h(\theta, x_1, \ldots, x_K; \phi)$
6:     $(\theta, \phi) \leftarrow (\theta, \phi) - \eta \nabla_{(\theta,\phi)} \sum_{k=1}^{K} \ell(g(x_k;\theta'), y_k)$
// Test time adaptation procedure
Require: $\theta$, $\phi$, test batch $x_1, \ldots, x_K$
7: $\theta' \leftarrow h(\theta, x_1, \ldots, x_K; \phi)$
8: $\hat{y}_k \leftarrow g(x_k; \theta')$ for $k = 1, \ldots, K$
```

We instantiate the model and adaptation procedure based on a contextual meta-learning approach with deep neural networks [69, 20]. In particular, we introduce two neural networks: a *context network* $f_{\text{cont}}(\cdot;\varphi) : \mathcal{X} \to \mathbb{R}^D$, parameterized by φ, and a *prediction network* $f_{\text{pred}}(\cdot,\cdot;\psi) : \mathcal{X} \times \mathbb{R}^D \to \mathcal{Y}$, parameterized by ψ. f_{cont} processes each example x_k in the mini batch separately to produce *contexts* $c_k \in \mathbb{R}^D$ for $k = 1, \ldots, K$, where D is a hyperparameter. In our experiments, we choose D to be the dimensionality of x. These contexts are averaged together into $\bar{c} = \frac{1}{K}\sum_{k=1}^{K} c_k$. f_{pred} similarly processes each x_k separately to produce an estimate of the output $\hat{y}_k$, but it additionally receives $\bar{c}$ as input. In this way, f_{cont} can provide information about the entire batch of K unlabeled data points to f_{pred} for predicting the correct outputs.

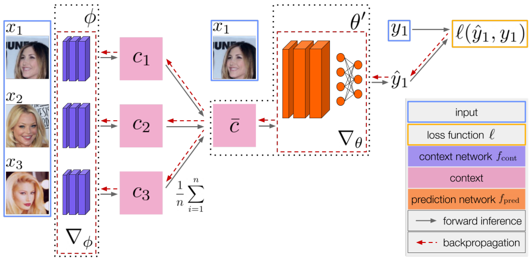


Figure 2: During inference, the context network produces a vector c_k for each input image x_k in the batch, and the average of these vectors $\bar{c}$ is input to the prediction network. The average context may adapt the model by providing helpful information about the underlying test distribution, e.g., the prevalence of certain groups such as blond-haired females, and this adaptation can aid prediction for difficult or ambiguous examples. During training, we compute the loss of the post adaptation predictions and backpropagate through the inference procedure to update our models.

A schematic of our approach is presented in Figure 2. The post adaptation model parameters θ' are $(\psi, \bar{c})$, whereas we can view the model parameters before adaptation as consisting of ψ and an undefined D-dimensional placeholder. Since we only ever use the model after adaptation, both during training and at test time, we can simply define $g(x;\theta') = f_{\text{pred}}(x, \bar{c}; \psi)$, leaving the model's behavior before adaptation undefined. We then also see that h is a function that takes in $(\psi, x_1, \ldots, x_K)$ and produces $\left(\psi, \frac{1}{K}\sum_{k=1}^{K} f_{\text{cont}}(x_k;\varphi)\right)$, and therefore its parameters ϕ are φ, the parameters of f_{cont}.

5 Experiments

Our experiments are designed to answer the following questions:

(1) Does our method for adaptive risk minimization learn models that can adapt to shift?

(2) How does our method for adaptation compare to group robustness methods?

(3) Can we visualize and understand scenarios where our method successfully adapts to shift?

5.1 Evaluation Domains and Protocol

We evaluate on three image classification benchmarks, which we choose because they incorporate meta-data that simplifies the process for constructing and testing against structural distribution shift, as described below. Additional experimental details are provided in Appendix A.

Rotated MNIST. We study a modified version of MNIST digit recognition [36] where images are rotated in 5 degree increments, for a total of 72 rotations. Taking each rotation to be a group, the smallest training groups, which consist of rotations from 0 to 115 degrees, contain only 135 data points each, whereas the entire training set contains 324000 points. At test time, we dynamically generate images from the MNIST test set with a certain rotation, and we consider each method's worst case and average accuracy across groups.

We compare our approach to (1) ERM, (2) distributionally robust neural networks (DRNN) [55], a state-of-the-art DRO method, and (3) a baseline that samples uniformly from each group. Sagawa et al. [55] refer to this baseline as upweighting (UW) and finds that it exhibits good worst case performance. We also consider a "context ablation" of our method that utilizes the same architecture as our method but samples uniformly from each group during training time, rather than sampling from tasks that induce distribution shift. This ablation helps determine whether our method benefits from this task-driven approach or if we simply benefit from observing multiple data points at a time.

Federated Extended MNIST (FEMNIST). Our second set of experiments uses the FEMNIST dataset [9], a version of the extended MNIST (EMNIST) dataset [11] that associates each handwritten character with the user that wrote the character. EMNIST consists of images of handwritten uppercase and lowercase letters, in addition to digits. We construct a training set of 62732 examples from 262 users, where the smallest user has 104 examples. The test set consists of 8439 examples from 35 users *not seen at training time*, and the smallest user has 140 examples. We measure each method's worst case and average test accuracy across users, as well as empirical accuracy on the test set.

We compare our method with ERM, the context ablation, and q-FedAvg [38], a state-of-the-art approach for fair federated learning that optimizes the q-fairness objective as described in Section 3. Similar to Sagawa et al. [55], Li et al. [38] also found that uniformly sampling from each user is a strong baseline. We thus also compare to this approach, which we refer to as UW for consistency.

For MNIST and FEMNIST, we use convolutional neural networks for both f_{cont} and f_{pred} in our method and the context ablation. The other methods also use convolutional networks. However, to be fair in terms of parameters, we increase the depth of the network used by these methods, such that it has *more* parameters than the total parameters of our context and prediction networks combined.

CelebA. CelebA is a dataset of celebrity faces with binary attributes attached to each photo [43]. We follow the protocol from Sagawa et al. [55], where the task corresponds to classifying the `Blond_Hair` attribute, and groups are constructed based on a combination of this attribute with the `Male` attribute, for a total of 4 groups. This pair of attributes is correlated in the training data, and the groups are strongly unbalanced, with the smallest group, corresponding to blond hair males, occupying only 1387 of the overall 162770 training data points.

In this domain, the label of interest is used to construct the groups, meaning that within a single group, all of the examples have the same label. Our method therefore has a distinct advantage when considering the worst case accuracy across groups, as the adaptation model in principle can update the model to output a constant label. We consider a more challenging evaluation scenario for our method: instead of measuring worst case accuracy over groups, we approximate the worst case accuracy over all *group distributions* using a binning strategy. Specifically, we divide the 3-dimensional probability simplex defined by the 4 groups into 5 bins: one bin each for one group occupying over 50% of the overall distribution, and one bin for when no groups occupy 50% of the distribution. During testing, we measure performance of our method on each bin separately and take the minimum across bins as

Significant improvement

Method	Worst Case Accuracy	Average Accuracy
ERM	0.8695 (0.0031)	0.9582 (0.0006)
UW baseline	0.8722 (0.0063)	0.9544 (0.0012)
DRNN [55]	0.8767 (0.0088)	0.9450 (0.0075)
Context ablation	0.7844 (0.0071)	0.9183 (0.0033)
ARM (ours)	**0.9372** (**0.0026**)	**0.9724** (**0.0013**)

Table 1: Comparison between ERM, UW, DRNN [55], the context ablation, and our method on worst case and average test accuracy across groups on the rotated MNIST dataset (standard error in parentheses). Our method significantly improves upon both metrics compared to all baselines, prior methods, and ablations.

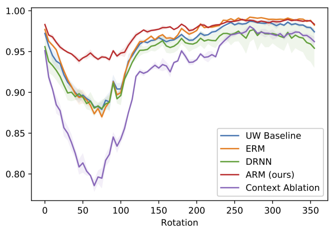


Figure 3: Test accuracy of each method as a function of the rotation. Our method performs significantly better on the rare groups.

Method	Worst Case Accuracy	Average Accuracy	Empirical Accuracy
ERM	0.6431 (0.0222)	0.8008 (0.0072)	0.8019 (0.0110)
UW baseline	0.6260 (0.0072)	0.8123 (0.0079)	0.8067 (0.0152)
q-FedAvg [38]	0.5819 (0.0098)	0.8075 (0.0034)	0.8118 (0.0019)
Context ablation	0.6436 (0.0073)	0.8149 (0.0068)	0.8177 (0.0056)
ARM (ours)	**0.6780** (**0.0133**)	**0.8570** (**0.0025**)	**0.8598** (**0.0036**)

Table 2: On the FEMNIST dataset, our method again improves upon all baselines, ablations, and prior methods in terms of all metrics, including a state-of-the-art method for fair federated learning [38]. The context ablation performs comparably to the other methods, though our method is still better.

the estimated worst case performance. This evaluation protocol consistently produces *lower* estimates of worst case performance for our method than simply evaluating across individual groups.

We again compare to ERM, DRNN, the UW baseline, and the context ablation. We largely follow the protocol from Sagawa et al. [55] when applicable: we use ResNet-50 models [25], pretrained on ImageNet, as the prediction model for all methods, with large weight decay and early stopping. However, we keep our context model as a simple convolutional neural network.

5.2 Quantitative Evaluation and Comparisons

Table 1 summarizes our results on the MNIST domain. We see that our method significantly improves worst case accuracy by over 6% on average compared to DRNN, which as expected provides the highest baseline for this metric. Unlike DRNN, our method does not sacrifice average accuracy in favor of robustness, as we also significantly improve in average accuracy compared to ERM, the highest baseline for this metric. We can see that our context ablation performs poorly, both compared to ARM, due to not explicitly training for adaptation, and compared to the other methods, which we attribute to having fewer parameters. In Figure 3, we visualize each method's accuracy as a function of the group. Though all methods perform worse on the rare groups, as seen by the consistent dip in performance, our method performs significantly better specifically on these groups, thus resulting in both better worst case accuracy and better average accuracy.

In Table 2, we summarize our results in the case of shift in the end user for the FEMNIST dataset. We again see that our method exhibits stronger results across all metrics compared to all baselines. Interestingly, we see that in our setting where train and test users don't overlap, q-FedAvg and UW perform worse in terms of worst case accuracy compared to ERM. We note that this prior method was previously only evaluated in the federated learning setting where each user's data is partitioned into a training and test set [38], and we believe our setting presents a harder problem that is representative of many real world settings. In order to gain intuition about the benefits of our method compared to ERM, we visualize predictions made by these models in subsection 5.3.

Finally, Table 3 summarizes the results of our approach and other methods on CelebA image classification. Our method consistently outperforms both DRNN and UW on the worst case, average, and empirical test accuracy metrics. This indicates that our method is able to successfully leverage unlabeled test data points to adapt the model and achieve greater robustness and performance. ERM achieves the strongest empirical test accuracy, however, consistent with the results from Sagawa et al. [55], we find that ERM has by far the lowest worst case accuracy. We note that our worst

Method	Worst Case Accuracy	Average Accuracy	Empirical Accuracy
ERM	0.4092 (0.0188)	0.8074 (0.0059)	**0.9533 (0.0001)**
UW baseline	0.8778 (0.0096)	0.9176 (0.0028)	0.9218 (0.0011)
DRNN [55]	0.8684 (0.0018)	0.9143 (0.0010)	0.9277 (0.0012)
Context ablation	0.8599 (0.0156)	0.9102 (0.0016)	0.9186 (0.0037)
ARM (ours)	**0.9098 (0.0016)**	**0.9237 (0.0007)**	0.9358 (0.0016)

Table 3: On the CelebA dataset, our method consistently attains better worst case accuracy than the other methods while also maintaining higher average and empirical accuracy. Note that we were able to significantly improve upon the UW results reported in Sagawa et al. [55], though our DRNN results are slightly worse. However, we note that our method also performs better than the DRNN results reported in Sagawa et al. [55].

case accuracy results for DRNN are slightly worse than those reported in Sagawa et al. [55] (0.8684 compared to 0.889). We attribute this difference to not using their group adjustment technique, however, our method still performs significantly better, achieving 0.9098 worst case accuracy. We present an additional experiment in Appendix B demonstrating even larger performance gains for our method when the prediction network is trained from scratch, rather than pretrained. Thus, we confirm our hypothesis that training models to adapt using unlabeled test batches leads to better overall performance compared to models trained for robustness.

5.3 Qualitative Analysis of Adaptive Risk Minimization

In Figure 4, we present an example of how our approach can improve test accuracy by adapting to specific users. We visualize a batch of 50 examples from a randomly sampled FEMNIST test user, and we highlight an ambiguous example. ERM and our method, when only given a batch size of 2 as shown by the black dashed box, incorrectly classify this example as "2". However, when given access to the entire batch of 50 images, which contain examples of class "2" and "a" from this user, our method successfully adapts this prediction to instead output "a", which is the correct label. In general, we find that most examples of adaptation in FEMNIST occur for similarly ambiguous examples, e.g., "l" versus "I", though not all examples were interpretable. In Appendix B, we plot performance as a function of the batch size at test time. Though our method is trained with batches of size 50, we find that the model is able to adapt with batch sizes as small as 10, indicating that our method can immediately begin to improve model performance even for small data set sizes.

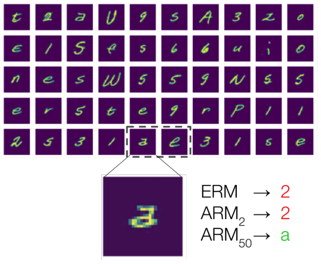


Figure 4: Visualizing one batch of 50 images from a test user in FEMNIST. Our method, using the entire batch, is able to successfully adapt to output the correct label "a" on the ambiguous example, shown enlarged, whereas ERM incorrectly outputs "2".

6 Discussion and Future Work

We presented adaptive risk minimization (ARM), a problem formulation for learning models that can robustly adapt in the face of group distribution shift at test time using only a batch of unlabeled test examples. We devised a method for optimizing the ARM objective that uses meta-learning to train models that are adaptable to sub-distributions of training data, thus not requiring information about the test distribution at training time, nor limiting the model to only one test distribution as in domain adaptation methods. Empirically, we demonstrated that our method improves performance in terms of both average and worst case metrics, as compared to ERM and group robustness approaches.

Currently, our method relies on ground truth meta-data for each training data point, similar to group robustness methods. Future work will aim to relieve this assumption, either by learning groupings of data or incorporating incomplete and partial meta-data into the learning process. Learning groups for robustness is a particularly interesting direction, as we hypothesize that properly learned groups may help in certain scenarios even when meta-data is available. For example, one might imagine grouping users together based on similarity in appearance, style, etc., rather than requiring robustness or adaptability over each individual user. Our method can also likely be further improved with more sophisticated meta-learning approaches, such as optimization based meta-learning using a learned loss function [2, 4], and this is another exciting direction for future work.

Broader Impact

Though machine learning systems have been deployed in many real world domains with great success, data that is anomalous or structurally different from the training data still sometimes renders these systems unreliable, harmful, or even dangerous. It is necessary, in order to realize the full potential of machine learning "in the wild", to have effective methods for detecting, robustifying against, and adapting to distribution shift. The potential upsides of developing such methods are clear. Imagine systems for image classification that fix incorrect or offensive outputs by adapting to each end user, or self driving cars that can smoothly adapt to driving in a new setting. We believe our work is a small step toward the goal of adapting in the face of distribution shift.

However, there are also complications and downsides that must be considered. For example, it is important to understand the failure modes and theoretical limits to handling distribution shift, otherwise we may place "false confidence" in our deployed systems, which may be catastrophic. Our work does not address this aspect of the problem, though this is an important direction for future work. Perhaps more insidiously, this line of research may grant even greater capabilities to parties that are able to collect larger and larger datasets. Deep learning systems are capable of effectively learning from ever growing data, and as the training data grows, the system can be trained to better adapt to a wider range of potential shifts. Thus, it is imperative to continue to push for high quality open source datasets, so that we may democratize the tools of machine learning.

Acknowledgements. MZ thanks Matt Johnson and Sharad Vikram for helpful discussions and is supported by an NDSEG fellowship. HM is funded by a scholarship from the Dr. Tech. Marcus Wallenberg Foundation for Education in International Industrial Entrepreneurship. AG is supported by an NSF graduate research fellowship. CF is a CIFAR Fellow in the Learning in Machines and Brains program. This research was supported by the DARPA Assured Autonomy and Learning with Less Labels programs.

References

[1] K. Allen, E. Shelhamer, H. Shin, and J. Tenenbaum. Infinite mixture prototypes for few-shot learning. In *International Conference on Machine Learning (ICML)*, 2019.

[2] A. Antoniou and A. Storkey. Learning to learn via self-critique. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[3] M. Baktashmotlagh, M. Harandi, B. Lovell, and M. Salzmann. Unsupervised domain adaptation by domain invariant projection. In *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[4] S. Bechtle, A. Molchanov, Y. Chebotar, E. Grefenstette, L. Righetti, G. Sukhatme, and F. Meier. Meta-learning via learned loss. *arXiv preprint arXiv:1906.05374*, 2019.

[5] A. Ben-Tal, D. den Hertog, A. De Waegenaere, B. Melenberg, and G. Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 2013.

[6] S. Bengio, Y. Bengio, J. Cloutier, and J. Gecsei. On the optimization of a synaptic learning rule. In *Optimality in Artificial and Biological Neural Networks*, 1992.

[7] J. Blanchet, Y. Kang, and K. Murthy. Robust Wasserstein profile inference and applications to machine learning. *arXiv preprint arXiv:1610.05627*, 2016.

[8] M. Buda, A. Maki, and M. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 2018.

[9] S. Caldas, S. Duddu, P. Wu, T. Li, J. Konečný, H. McMahan, V. Smith, and A. Talwalkar. LEAF: A benchmark for federated settings. In *Workshop on Federated Learning for Data Privacy and Confidentiality*, 2019.

[10] F. Chen, M. Luo, Z. Dong, Z. Li, and X. He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018.

[11] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. EMNIST: An extension of MNIST to handwritten letters. *arXiv preprint arXiv:1702.05373*, 2017.

[12] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh. Sample selection bias correction theory. In *International Conference on Algorithmic Learning Theory (ALT)*, 2008.

[13] G. Csurka. Domain adaptation for visual applications: A comprehensive survey. *arXiv preprint: arXiv:1702.05374*, 2017.

[14] Q. Dou, D. Castro, K. Kamnitsas, and B. Glocker. Domain generalization via model-agnostic learning of semantic features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[15] J. Duchi, P. Glynn, and H. Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *arXiv preprint arXiv:1610.03425*, 2016.

[16] P. Esfahani and D. Kuhn. Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations. *arXiv preprint arXiv:1505.05116*, 2015.

[17] A. Fallah, A. Mokhtari, and A. Ozdaglar. Personalized federated learning: A meta-learning approach. *arXiv preprint arXiv:2002.07948*, 2020.

[18] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning (ICML)*, 2017.

[19] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning (ICML)*, 2015.

[20] M. Garnelo, D. Rosenbaum, C. Maddison, T. Ramalho, D. Saxton, M. Shanahan, Y. Teh, D. Rezende, and S. Eslami. Conditional neural processes. In *International Conference on Machine Learning (ICML)*, 2018.

[21] A. Globerson and S. Roweis. Nightmare at test time: Robust learning by feature deletion. In *International Conference on Machine Learning (ICML)*, 2006.

[22] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] A. Gupta, B. Eysenbach, C. Finn, and S. Levine. Unsupervised meta-learning for reinforcement learning. *arXiv preprint arXiv:1806.04640*, 2018.

[24] J. Harrison, A. Sharma, C. Finn, and M. Pavone. Continuous meta-learning without tasks. *arXiv preprint arXiv:1912.08866*, 2019.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[26] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations (ICLR)*, 2019.

[27] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019.

[28] S. Hochreiter, A. Younger, and P. Conwell. Learning to learn using gradient descent. In *International Conference on Artificial Neural Networks (ICANN)*, 2001.

[29] S. Horiguchi, S. Amano, M. Ogawa, and K. Aizawa. Personalized classifier for food image recognition. *IEEE Transactions on Multimedia*, 2018.

[30] K. Hsu, S. Levine, and C. Finn. Unsupervised learning via meta-learning. In *International Conference on Learning Representations (ICLR)*, 2019.

[31] W. Hu, G. Niu, I. Sato, and M. Sugiyama. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning (ICML)*, 2018.

[32] J. Huang, A. Smola, A. Gretton, K. Borgwardt, and B. Schölkopf. Correcting sample selection bias by unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[33] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

[34] Y. Jiang, J. Konečný, K. Rush, and S. Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[35] D. Lazer, R. Kennedy, G. King, and A. Vespignani. The parable of Google flu: Traps in big data analysis. *Science*, 2014.

[36] Y. LeCun, C. Cortes, and C. Burges. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`. Accessed: 2020/06/01.

[37] D. Li, Y. Yang, Y. Song, and T. Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[38] T. Li, M. Sanjabi, A. Beirami, and V. Smith. Fair resource allocation in federated learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[39] X. Li, Q. Sun, Y. Liu, Q. Zhou, S. Zheng, T. Chua, and B. Schiele. Learning to self-train for semi-supervised few-shot classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[40] S. Lin, Y. Guang, and J. Zhang. Real-time edge intelligence in the making: A collaborative learning framework via federated meta-learning. *arXiv preprint arXiv:2001.03229*, 2020.

[41] Z. Lipton, Y. Wang, and A. Smola. Detecting and correcting for label shift with black box predictors. In *International Conference on Machine Learning (ICML)*, 2018.

[42] A. Liu and B. Ziebart. Robust classification under sample selection bias. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[43] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[44] M. Long, Z. Cao, J. Wang, and M. Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[45] H. McMahan, E. Moore, D. Ramage, S. Hampson, and B. Arcas. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[46] L. Metz, N. Maheswaranathan, B. Cheung, and J. Sohl-Dickstein. Meta-learning update rules for unsupervised representation learning. In *International Conference on Learning Representations (ICLR)*, 2019.

[47] T. Miyato, S. Maeda, M. Koyama, K. Nakae, and S. Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015.

[48] M. Mohri, G. Sivek, and A. Suresh. Agnostic federated learning. In *International Conference on Machine Learning (ICML)*, 2019.

[49] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[50] Y. Oren, S. Sagawa, T. Hashimoto, and P. Liang. Distributionally robust language modeling. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2019.

[51] J. Quiñonero Candela, M. Sugiyama, A. Schwaighofer, and N. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.

[52] S. Ravi and H. Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[53] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. Tenenbaum, H. Larochelle, and R. Zemel. Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations (ICLR)*, 2015.

[54] A. Royer and C. Lampert. Classifier adaptation at prediction time. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[55] S. Sagawa, P. Koh, T. Hashimoto, and P. 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.

[56] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning (ICML)*, 2016.

[57] J. Schmidhuber. Evolutionary principles in self-referential learning. *Diploma thesis, Institut f. Informatik, Tech. Univ. Munich*, 1987.

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

[59] S. Shan and J. Oliva. Meta-neighborhoods. *arXiv preprint arXiv:1909.09140*, 2019.

[60] L. Shen, Z. Lin, and Q. Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *European Conference on Computer Vision (ECCV)*, 2016.

[61] H. Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference (JSPI)*, 2000.

[62] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[63] A. Storkey and M. Sugiyama. Mixture regression for covariate shift. In *Advances in Neural Information Processing Systems (NIPS)*, 2007.

[64] M. Sulc and J. Matas. Improving CNN classifiers by estimating test-time priors. In *IEEE International Conference on Computer Vision (ICCV)*, 2019.

[65] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt. Test-time training for out-of-distribution generalization. *arXiv preprint arXiv:1909.13231*, 2019.

[66] S. Thrun and L. Pratt. *Learning to Learn*. Springer Science & Business Media, 1998.

[67] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[68] V. Vapnik. *Statistical Learning Theory*. Wiley New York, 1998.

[69] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[70] Y. Wang, D. Ramanan, and M. Hebert. Learning to model the tail. In *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[71] T. Yu, C. Finn, A. Xie, S. Dasari, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *RSS*, 2018.

[72] M. Zafar, I. Valera, M. Rodriguez, and K. Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *International World Wide Web Conference (WWW)*, 2017.

A Additional Experimental Details

When reporting our results, we run each method across three seeds and report the mean and standard error across seeds. Standard error is calculated as the sample standard deviation divided by the square root of 3. We checkpoint models after every epoch of training, and at test time, we evaluate the checkpoint with the best worst case validation accuracy. Training hyperparameters and details for how we evaluate validation and test accuracy are provided for each experimental domain below.

A.1 Rotated MNIST Details

We construct a training set of 324000 data points by replicating 90% of the original training set – separating out a validation set – 6 times and applying random rotations to each image. The rotations are not dependent on the image or label, but certain rotations are sampled much less frequently than others. In particular, rotations of 0 through 115 degrees, inclusive, have 135 data points each, 120 through 235 degrees have 1350 points each, and 240 through 355 degrees have 12015 points each.

We train all models for 200 epochs with mini batch sizes of 50. We sample uniformly across rotations as this is a standard technique for improving performance on rare groups [8, 55]. We use Adam updates with learning rate 0.0001 and weight decay 0.0001. We construct an additional level of mini batching for our method as described in subsection 4.2, such that the batch dimensions of the data mini batches is 6×50 rather than just 50, and each of the inner mini batches contain examples from the same rotation. We refer to the outer batch dimension as the *meta batch size* and the inner dimension as the batch size. All methods are still trained for the same number of epochs and see the same amount of data. Finally, DRNN uses an additional learning rate hyperparameter for their robust loss, which we set to 0.01 across all experiments [55].

Due to the large number of groups in this setting, we only compute validation accuracy every 10 epochs. When computing validation accuracy, we estimate accuracy on each rotation by randomly sampling 300 of the held out 6000 original training points and applying the specific rotation, resampling for each validation evaluation. This is effectively the same procedure as the test evaluation, which randomly samples 3000 of the 10000 test points and applies a specific rotation.

We retain the original $28 \times 28 \times 1$ dimensionality for the MNIST images, and we divide inputs by 256. We use convolutional neural networks for all methods with varying depths to account for parameter fairness. For ERM, the UW baseline, and DRNN, the network has four convolution layers with 128 filters of size 5×5, followed by 4×4 average pooling, one fully connected layer of size 200, and a linear output layer. Rectified linear unit (ReLU) nonlinearities are used throughout, and batch normalization [33] is used for the convolution layers. The first two convolution layers use padding to preserve the input height and width, and the last two convolution layers use 2×2 max pooling. For our method and context ablation, we remove the first two convolution layers for the prediction network, but we incorporate a context network. The context network uses two convolution layers with 64 filters of size 5×5, with ReLU nonlinearities, batch normalization, and padding, followed by a final convolution layer with padding. This last layer has number of filters, of size 5×5, equal to the channel dimensionality of the input, which in this case is 1.

A.2 FEMNIST Details

FEMNIST, and EMNIST in general, is a significantly more challenging dataset compared to MNIST due to its larger label space (62 compared to 10 classes), label imbalance (almost half of the data points are digits), and inherent ambiguities (e.g., lowercase versus uppercase “o”) [11]. In processing the FEMNIST dataset,[2] we filter out users with fewer than 100 examples, leaving 262, 50, and 35 unique users and a total of 62732, 8484, and 8439 data points in the training, validation, and test splits, respectively. The smallest users contain 104, 119, and 140 data points, respectively. We keep all hyperparameters the same as MNIST, except we set the meta batch size for our method to be 2.

We compare to q-FedAvg rather than DRNN on this domain, as this method is specifically designed for federated learning settings [38]. We modify the authors’ publicly available code[3] to run experiments in our setting, and we will make this fork available upon publication along with

[2] Obtained from `https://github.com/TalwalkarLab/leaf/tree/master/data/femnist`.

[3] `https://github.com/litian96/fair_flearn`

our own code base. This method follows its own update rule and hyperparameter settings, and we separately optimize the hyperparameters for q-FedAvg as described in Li et al. [38]. Specifically, we first set $q = 0$ and sweep learning rate values between 0.0001 and 1.0, and then we sweep $q \in \{0.001, 0.01, 0.1, 0.5, 1, 2, 5, 10, 15\}$ with the optimal learning rate. With this procedure, we set learning rate to be 0.8 and q to be 0.001.

We compute validation accuracy every epoch by iterating through the data of each validation user once, and this procedure is the same as test evaluation. Note that all methods will sometimes receive small batch sizes as each user's data size may not be a multiple of 50, and though this may affect our method and the context ablation, we demonstrate in Appendix B that our method can adapt using batch sizes much smaller than 50. The network architectures are the same as the architectures used for rotated MNIST.

A.3 CelebA Details

We obtain the CelebA dataset from Kaggle.[4] We retain the standard train, validation, and test splits, which have 162770, 19867, and 19962 data points, respectively. There are four groups from the combination of the `Blond_Hair` attribute, which is also the label, and the `Male` attribute. The smallest group across the training, validation, and test splits corresponds to blond hair males, with 1387, 182, and 180 data points, respectively.

We run two separate sets of experiments on this domain. First, as reported in subsection 5.2, we use a ResNet-50 [25] model pretrained on ImageNet, which we obtain from the `torchvision` library of pretrained models. This aligns with the experimental protocol followed in Sagawa et al. [55]. We additionally run experiments using the same model trained from scratch, and we report results for this setting in Appendix B. We describe all hyperparameter differences between the two settings below – unless otherwise stated, other hyperparameters are identical to those in the FEMNIST experiments.

First, we train for 50 epochs, when we use the pretrained network, or 100 epochs when training from scratch. Second, we set weight decay to 0.5. This weight decay hyperparameter deviates from traditional deep learning practice but has been shown by Sagawa et al. [55] to be helpful for achieving good generalization on the worst case group, thus we adopt this approach as well. Third, we decay the learning rate to 0.00001 after one training epoch, which Sagawa et al. [55] use throughout training. Lastly, for our method, each inner batch consists of examples from the same *distribution* over groups, rather than from the same group. Specifically, we construct a flat Dirichlet distribution, i.e., with concentration parameter $\boldsymbol{\alpha}$ as a 4-dimensional vector of ones. For each training step, we sample from this distribution twice, when using the pretrained network, or 3 times, if training from scratch. We then use these samples as the parameters of the categorical distribution over groups for sampling 50 training examples each, again giving us batch size 50 and meta batch size either 2 or 3. This sampling procedure at training time matches the validation and test evaluation, which we perform over distributions of groups rather than groups themselves.

Though the prediction network is a ResNet-50, we keep the context network as the same simple convolutional network as in the previous experiments, except the output channel dimension is changed to 3 in line with the input dimensionality. For data preprocessing we apply a 178×178 center crop to each image and normalize using the ImageNet mean and standard deviation. When using the pretrained network, which expects a particular input dimensionality, we further resize the input to $224 \times 224 \times 3$. Finally, we compute the worst case validation accuracy using the same binning strategy for computing worst case test accuracy. Specifically, we sample 15 distributions uniformly from each bin, sample 1500 images per distribution, and average the accuracies per bin before taking measuring the minimum across bins. For test evaluation, we sample 3000 images per distribution.

B Additional Experiments

B.1 CelebA without ImageNet Pretraining

Table 4 presents our results on the CelebA dataset when training from scratch. We see that accuracy in general suffers from not having access to ImageNet pretraining, and our model again consistently beats out the UW baseline and DRNN on all metrics. There is a noticeably larger gap specifically in

[4] `https://www.kaggle.com/jessicali9530/celeba-dataset/data`

Method	Worst Case Accuracy	Average Accuracy	Empirical Accuracy
UW baseline	0.8297 (0.0145)	0.8865 (0.0035)	0.8953 (0.0043)
DRNN [55]	0.8426 (0.0122)	0.8902 (0.0029)	0.8980 (0.0008)
ARM (ours)	**0.8900 (0.0053)**	**0.9016 (0.0018)**	**0.9269 (0.0069)**

Table 4: On the CelebA dataset, when training from scratch, we see lower accuracies across the board as expected. However, our method shows even more significant gains over the UW baseline and DRNN in terms of worst case accuracy across groups, coming closer to its performance when using the pretrained model. We omit ERM, due to its poor worst case accuracy, and the context ablation, due to its worse performance on all metrics compared to the three methods considered here, though we note that ERM would likely still have the highest empirical accuracy in this setting.

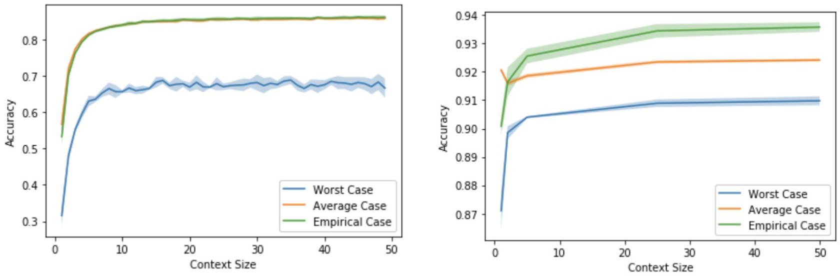


Figure 5: Worst case (blue), average (orange), and empirical (green) performance of ARM on FEMNIST (left) and CelebA (right) as a function of the test batch size. We do not measure performance for every test batch size for CelebA for computational efficiency. For FEMNIST, average and empirical performance track each other very closely, though the difference is more pronounced on CelebA due to the larger difference in group sizes. For both domains, we see that our method can adapt using batch sizes as small as 10, despite training with batches of size 50.

terms of worst case accuracy compared to the previous results that were obtained using pretrained models, with our method coming closest to its previous results. We note again that DRNN may enjoy a slight performance boost compared to our reported results if we further include the group adjustment technique from Sagawa et al. [55], though we believe it is unlikely that this technique would fully close the gap between DRNN and our method.

B.2 Test Batch Size and Adaptation Accuracy

Figure 5 visualizes the performance of our method as a function of the batch size at test time. We only change the test batch size in these plots, and the models were trained with batch sizes of 50. We can see that adaptation performance starts to improve sometimes with batches as small as 2 images, and performance is close to the original accuracy with batches as small as 10 images. This lends support to our hypothesis that our method can successfully adapt to test time shifts, without access to any test data labels and even with small data sizes.