Learning Stable Classifiers by Transferring Unstable Features

Yujia Bao [1] **Shiyu Chang** [2] **Regina Barzilay** [1]

Abstract

While unbiased machine learning models are essential for many applications, bias is a human-defined concept that can vary across tasks. Given only input-label pairs, algorithms may lack sufficient information to distinguish stable (causal) features from unstable (spurious) features. However, related tasks often share similar biases – an observation we may leverage to develop stable classifiers in the transfer setting. In this work, we explicitly inform the target classifier about unstable features in the source tasks. Specifically, we derive a representation that encodes the unstable features by contrasting different data environments in the source task. We achieve robustness by clustering data of the target task according to this representation and minimizing the worst-case risk across these clusters. We evaluate our method on both text and image classifications. Empirical results demonstrate that our algorithm is able to maintain robustness on the target task for both synthetically generated environments and real-world environments. Our code is available at `https://github.com/YujiaBao/Tofu`.

1. Introduction

Automatic de-biasing (Sohoni et al., 2020; Creager et al., 2021; Sanh et al., 2021) has emerged as a promising direction for learning stable classifiers. The key premise here is that no additional annotations for the bias attribute are required. However, bias is a human-defined concept and can vary from task to task. Provided with only input-label pairs, algorithms may not have sufficient information to distinguish stable (causal) features from unstable (spurious) features.

To address this challenge, we note that related tasks are often fraught with similar spurious correlations. For instance, when classifying animals such as camels vs. cows, their backgrounds (desert vs. grass) may constitute a spurious correlation (Beery et al., 2018). The same bias between the label and the background also persists in other related classification tasks (such as sheep vs. antelope). In the resource-scarce target task, we only have access to the input-label pairs. However, in the source tasks, where training data is sufficient, identifying biases may be easier. For instance, we may have examples collected from multiple environments, in which correlations between bias features and the label are different (Arjovsky et al., 2019). These source environments help us define the exact bias features that we want to regulate.

One obvious approach to utilize the source task is direct transfer. Specifically, given multiple source environments, we can train an unbiased source classifier and then apply its representation to the target task. However, we empirically demonstrate that while the source classifier is not biased when making its final predictions, its internal continuous representation can still encode information about the unstable features. Figure 1 shows that in Colored MNIST, where the digit label is spuriously correlated with the image color, direct transfer by either re-using or fine-tuning the representation learned on the source task fails in the target task, performing no better than the majority baseline.

In this paper, we propose to explicitly inform the target classifier about unstable features from the source data. Specifically, we derive a representation that encodes these unstable features using the source environments. Then we identify distinct subpopulations by clustering examples based on this representation and apply group DRO (Sagawa et al., 2019) to minimize the worst-case risk over these subpopulations. As a result, we enforce the target classifier to be robust against different values of the unstable features. In the example above, animals would be clustered according to backgrounds, and the classifier should perform well regardless of the clusters (backgrounds).

The remaining question is how to compute the unstable feature representation using the source data environments. Following Bao et al. (2021), we hypothesize that unstable features are reflected in mistakes observed during classifier transfer across environments. For instance, if the classifier uses the background to distinguish camels from cows, the

[1]MIT CSAIL [2]Computer Science, UC Santa Barbara. Correspondence to: Yujia Bao <yujia@csail.mit.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

Figure 1: Transferring across tasks in Colored MNIST (Arjovsky et al., 2019). On the source task, we learn a color-invariant model that achieves oracle performance (given direct access to the unstable features). However, directly transferring this model to the target task, by reusing or fine-tuning its feature extractor, severely overfits the spurious correlation and underperforms the majority baseline (50%) on a test set where the spurious correlation flips. By explicitly transferring the unstable features, our algorithm TOFU (Transfer OF Unstable features) is able to reach the oracle performance.

camel images that are predicted correctly would have a desert background while those predicted incorrectly are likely to have a grass background. More generally, we prove that among examples with the same label value, those with the same prediction outcome will have more similar unstable features than those with different predictions. By forcing examples with the same prediction outcome to stay closer in the feature space, we obtain a representation that encodes these latent unstable features.

We evaluate our approach, Transfer OF Unstable features (TOFU), on both synthetic and real-world environments. Our synthetic experiments first confirm our hypothesis that standard transfer approaches fail to learn a stable classifier for the target task. By explicitly transferring the unstable features, our method significantly improves over the best baseline across 12 transfer settings (22.9% in accuracy), and reaches the performance of an oracle model that has direct access to the unstable features (0.3% gap). Next, we consider a practical setting where environments are defined by an input attribute and our goal is to reduce biases from other unknown attributes. On CelebA, TOFU achieves the best worst-group accuracy across 38 latent attributes, outperforming the best baseline by 18.06%. Qualitative and quantitative analyses confirm that TOFU is able to identify the unstable features.

2. Related work

Removing bias via annotations: Due to idiosyncrasies of the data collection process, annotations are often coupled with unwanted biases (Buolamwini & Gebru, 2018; Schuster et al., 2019; McCoy et al., 2019; Yang et al., 2019). To address this issue and learn robust models, researchers leverage extra information (Belinkov et al., 2019; Stacey et al., 2020; Hinton, 2002; Clark et al., 2019; He et al., 2019; Mahabadi et al., 2019). One line of work assumes that the bias attributes are known and have been annotated for each example, e.g., group distributionally robust optimization (DRO) (Hu et al., 2018; Oren et al., 2019; Sagawa et al., 2020). By defining groups based on these bias attributes, we explicitly specify the distribution family to optimize over. However, identifying the hidden biases is time-consuming and often requires domain knowledge (Zellers et al., 2019; Sakaguchi et al., 2020). To address this issue, another line of work (Peters et al., 2016; Krueger et al., 2020; Chang et al., 2020; Jin et al., 2020; Ahuja et al., 2020; Arjovsky et al., 2019; Bao et al., 2021; Kuang et al., 2020; Shen et al., 2020) only assumes access to a set of data environments. These environments are defined based on readily-available information of the data collection circumstances, such as location and time. The main assumption is that while spurious correlations vary across different environments, the association between the causal features and the label should stay the same. Thus, by learning a representation that is invariant across all environments, they alleviate the dependency on spurious features. In contrast to previous works, we don't have access to any additional information besides the labels in our target task. We show that we can achieve robustness by transferring the unstable features from a related source task.

Automatic de-biasing A number of recent approaches focus on a more common setting where the algorithm only has access to the input-label pairs. Li & Xu (2021) leverages disentangled representations in generative models to identify biases. Sanh et al. (2021); Nam et al. (2020); Utama et al. (2020) find that weak models are more vulnerable to spurious correlations as they only learn shallow heuristics. By boosting from their mistakes, they obtain a more robust model. Qiao et al. (2020) uses adversarial learning to augment the biased training data. Creager et al. (2021); Sohoni et al. (2020); Ahmed et al. (2020); Matsuura & Harada (2020); Liu et al. (2021) propose to identify minority groups by looking at the features produced by a biased model.

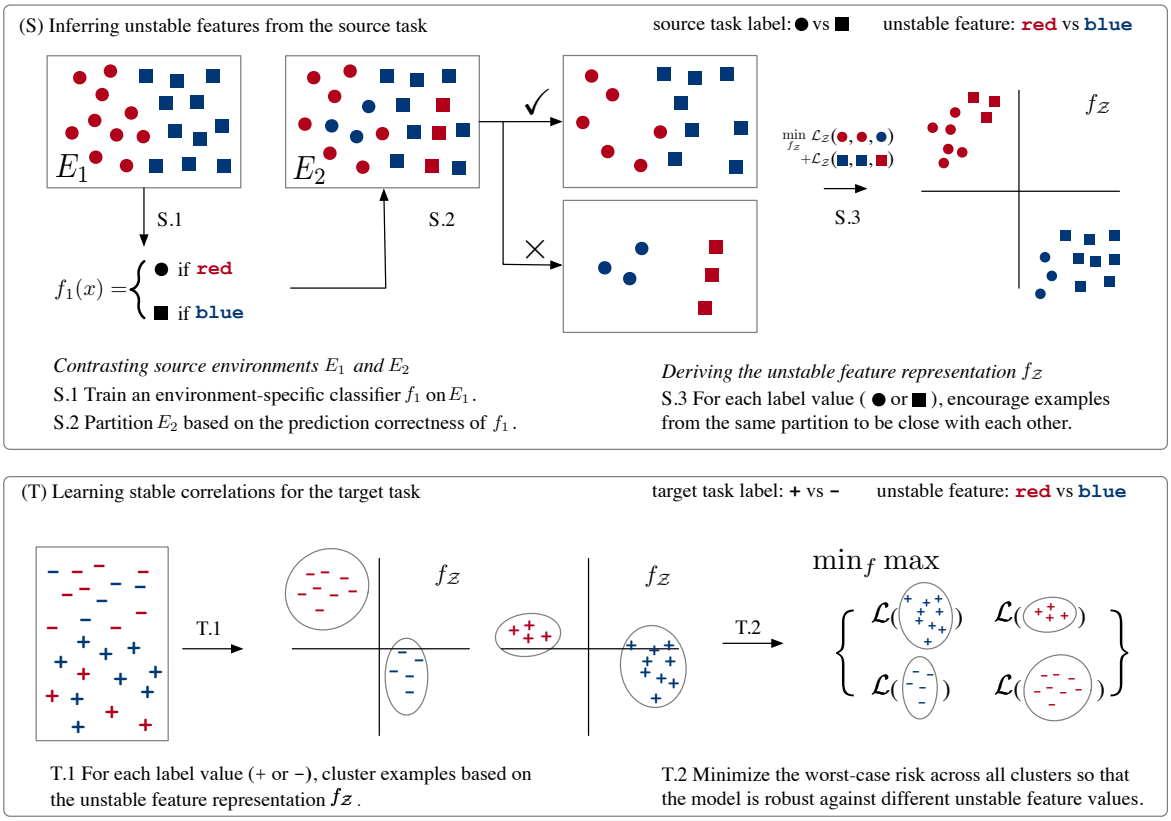

Figure 2: Our algorithm TOFU 1) infers unstable features from the source task (Section 3.1) and 2) learns stable correlations for the target task (Section 3.2). We create partitions for all environment pairs. For ease of illustration, we only depict using f_1 to partition E_2. Best viewed in color.

These automatic approaches are intriguing as they do not require additional annotation. However, we note that bias is a human-centric concept and can vary from tasks to tasks. For models that only have access to the input-label pairs, they have no information to distinguish causal features from bias features. For example, consider the Colored MNIST dataset, where color and digit shape are correlated in the training set but not in the test set. If our task is to predict the digit, then color becomes the spurious bias that we want to remove. Vice versa, if we want to predict the color, then digit shape is spurious. Creager et al. (2021); Nam et al. (2020) empirically demonstrate that they can learn a color-invariant model for the digit prediction task. However, their approaches will result in the same color-invariant model for the color prediction task, and thus fail at test time, when color and digit are no longer correlated. In this work, we leverage source tasks to define the exact bias that we want to remove for the target task.

Transferring robustness across tasks: Prior work has also studied the transferability of adversarial robustness across tasks. For example, Hendrycks et al. (2019); Shafahi et al. (2020) show that by pre-training the model on a large-scale source task, we can improve the model robustness against adversarial perturbations over l_∞ norm. We note that these perturbations measure the smoothness of the classifier, rather than the stability of the classifier against spurious correlations. In fact, our results show that if we directly re-use or fine-tune the pre-trained feature extractor on the target task, the model will quickly over-fit to the unstable correlations present in the data. We propose to address this issue by explicitly inferring the unstable features using the source environments and use this information to guide the target classifier during training.

3. Method

Problem formulation We consider the transfer problem from a source task to a target task. For the source task, we assume the standard setting (Arjovsky et al., 2019) where the training data contain n environments $E_1, \ldots, E_n$. Within each environment E_i, examples are drawn from the joint distribution $P_i(x, y)$. Following Woodward (2005), we define unstable features $\mathcal{Z}(x)$ as features that are *differentially* correlated with the label across the environments. We note that $\mathcal{Z}(x)$ is unknown to the model.

For the target task, we only have access to the input-label pairs (x, y) (i.e. no environments). We assume that the target label is *not* causally associated with the above unstable features $\mathcal{Z}$. However, due to collection biases, the target data may contain *spurious correlations* between the label and $\mathcal{Z}$. Our goal is to transfer the knowledge that $\mathcal{Z}$ is unstable in the source task, so that the target classifier will not rely on these spurious features.

Overview If the unstable features have been identified for the target task, we can simply apply group DRO to learn a stable classifier. By grouping examples based on the unstable features and minimizing the worst-case risk over these *manually-defined* groups, we explicitly address the bias from these unstable features (Hu et al., 2018; Oren et al., 2019; Sagawa et al., 2020). In our setup, while these unstable features are not accessible, we can leverage the source environments to derive groups over the target data that are informative of these biases. Applying group DRO on these *automatically-derived* groups, we can eliminate the unstable correlations in the target task.

Our overall transfer paradigm is depicted in Figure 2. It consists of two steps: inferring unstable features from the source task (Section 3.1) and learning stable correlations for the target task (Section 3.2). First, for the source task we use a classifier trained on one environment to partition data from another environment based on the correctness of its predictions. Starting from the theoretical results in (Bao et al., 2021), we show that these partitions reflect the similarity of the examples in terms of their unstable features: among examples with the same label value, those that share the same prediction outcome have more similar unstable features than those with different predictions (Theorem 1). We can then derive a representation $f_{\mathcal{Z}}$ where examples are distributed based on the unstable features $\mathcal{Z}$. Next, we cluster target examples into groups based on the learned unstable feature representation $f_{\mathcal{Z}}$. These *automatically-derived* groups correspond to different modes of the unstable features, and they act as proxies to the *manually-defined* groups in the oracle setting where unstable features are explicitly annotated. Finally, we use group DRO to obtain our robust target classifier by minimizing the worst-case risk over these groups.

3.1. Inferring unstable features from the source task

Given the data environments from the source task, we would like to 1) identify the unstable correlations across these environments; 2) learn a representation $f_{\mathcal{Z}}(x)$ that encodes the unstable features $\mathcal{Z}(x)$. We achieve the first goal by contrasting the empirical distribution of different environments (Figure 2.S.1 and Figure 2.S.2) and the second goal by metric learning (Figure 2.S.3).

Let E_i and E_j be two different data environments. Bao et al. (2021) shows that by training a classifier f_i on E_i and using it to make predictions on E_j, we can reveal the unstable correlations from its prediction results. Intuitively, if the unstable correlations are stronger in E_i, the classifier f_i will overuse these correlations and make mistakes on E_j when these stronger correlations do not hold.

In this work, we connect the prediction results directly to the unstable features. We show that the prediction results of the classifier f_i on E_j estimate the relative distance of the unstable features.

Theorem 1 (Simplified)**.** *Consider examples in E_j with label value y. Let $X_1^{\checkmark}, X_2^{\checkmark}$ denote two batches of examples that f_i predicted correctly, and let $X_3^{\times}$ denote a batch of incorrect predictions. We use $\overline{\cdot}$ to represent the mean across a given batch. Following the same assumption in (Bao et al., 2021), we have*

$$\|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_2^{\checkmark})\|_2 < \|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_3^{\times})\|_2$$

almost surely for large enough batch size.[1]

The result makes intuitive sense as we would expect example pairs that share the same prediction outcome should be more similar than those with different prediction outcomes. We note that it is critical to look at examples with the same label value; otherwise, the unstable features will be coupled with the task-specific label in the prediction results.

While the value of the unstable features $\mathcal{Z}(x)$ is still not directly accessible, Theorem 1 enables us to learn a feature representation $f_{\mathcal{Z}}(x)$ that preserves the distance between the examples in terms of their unstable features. We adopt standard metric learning (Chechik et al., 2010) to minimize the following triplet loss:

$$\mathcal{L}_{\mathcal{Z}}(X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}) = \max(0, \delta + \|\overline{f_{\mathcal{Z}}}(X_1^{\checkmark}) - \overline{f_{\mathcal{Z}}}(X_2^{\checkmark})\|_2^2 - \|\overline{f_{\mathcal{Z}}}(X_1^{\checkmark}) - \overline{f_{\mathcal{Z}}}(X_3^{\times})\|_2^2), \tag{1}$$

where δ is a hyper-parameter. By minimizing Eq (1), we encourage examples that have similar unstable features to be close in the representation $f_{\mathcal{Z}}$. To summarize, inferring unstable features from the source task consists of three steps (Figure 2.S):

S.1 For each source environment E_i, train an environment-specific classifier f_i.

S.2 For each pair of environments E_i and E_j, use classifier f_i to partition E_j into two sets: $E_j^{i\checkmark}$ and $E_j^{i\times}$, where $E_j^{i\checkmark}$ contains examples that f_i predicted correctly and $E_j^{i\times}$ contains those predicted incorrectly.

[1] See Appendix B for the full theorem and proof.

S.3 Learn an unstable feature representation $f_{\mathcal{Z}}$ by minimizing Eq (1) across all pairs of environments E_i, E_j and all possible label value y:

$$f_{\mathcal{Z}} = \arg\min \sum_{y, E_i \neq E_j} \mathbb{E}_{X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}} \left[\mathcal{L}_{\mathcal{Z}}(X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times})\right]$$

where batches $X_1^{\checkmark}, X_2^{\checkmark}$ are sampled uniformly from $E_j^{i\checkmark}|_y$ and batch $X_3^{\times}$ is sampled uniformly from $E_j^{i\times}|_y$ ($\cdot|_y$ denotes the subset of $\cdot$ with label value y).

3.2. Learning stable correlations for the target task

Given the unstable feature representation $f_{\mathcal{Z}}$, our goal is to learn a target classifier that focuses on the stable correlations rather than using unstable features. Inspired by group DRO (Sagawa et al., 2020) we minimize the worst-case risk across groups of examples that are representative of different unstable feature values. However, in contrast to DRO, these groups are constructed automatically based on the previously learned representation $f_{\mathcal{Z}}$.

For each target label value y, we use the representation $f_{\mathcal{Z}}$ to cluster target examples with label y into different clusters (Figure 2.T.1). Since these clusters capture different modes of the unstable features, they are approximations of the typical manually-defined groups when annotations of the unstable features are available. By minimizing the worst-case risk across all clusters, we explicitly enforce the classifier to be robust against unstable correlations (Figure 2.T.2). We note that it is important to cluster within examples of the same label, as opposed to clustering the whole dataset. Otherwise, the cluster assignment may be correlated with the target label.

Concretely, learning stable correlations for the target task has two steps (Figure 2.T).

T.1 For each label value y, apply K-means (l_2 distance) to cluster examples with label y in the feature space $f_{\mathcal{Z}}$. We use $C_1^y, \ldots, C_{n_c}^y$ to denote the resulting cluster assignment, where n_c is a hyper-parameter.

T.2 Train the target classifier f by minimizing the worst-case risk over all clusters:

$$f = \arg\min \max_{i,y} \mathcal{L}(C_i^y),$$

where $\mathcal{L}(C_i^y)$ is the empirical risk on cluster C_i^y.

4. Experimental setup

4.1. Datasets and settings

Synthetic environments We start with controlled experiments where environments are created based on the spurious correlation. We consider four datasets: MNIST (LeCun et al., 1998), BeerReview (McAuley et al., 2012), ASK2ME (Bao et al., 2019a) and Waterbird (Sagawa et al., 2019). In MNIST and BeerReview, we inject spurious feature to the input (background color for MNIST and pseudo token for BeerReview). In ASK2ME and Waterbird, spurious feature corresponds to an attribute of the input (`breast_cancer` for ASK2ME and `background` for Waterbird).

Table 1: Pearson correlation coefficient between the spurious feature $\mathcal{Z}$ and the label Y for each task. The validation environment E^{val} follows the same distribution as E_1^{train}. We study the transfer problem between different task pairs. For the source task S, the model can access $E_1^{\text{train}}(S), E_2^{\text{train}}(S)$ and $E^{\text{val}}(S)$. For the target task T, the model can access $E_1^{\text{train}}(T)$ and $E^{\text{val}}(T)$.

$\rho(\mathcal{Z}, Y)$	Task	E_1^{train}	E_2^{train}	E^{val}	E^{test}
MNIST	ODD	0.87	0.75	0.87	-0.11
	EVEN	0.87	0.75	0.87	-0.11
BEER REVIEW	LOOK	0.60	0.80	0.60	-0.80
	AROMA	0.60	0.80	0.60	-0.80
	PALATE	0.60	0.80	0.60	-0.80
ASK2ME	PENE.	0.31	0.52	0.31	0.00
	INCI.	0.44	0.66	0.44	0.00
WATERBIRD	WATER	0.36	0.63	0.36	0.00
	SEA	0.39	0.64	0.39	0.00

For each dataset, we consider multiple tasks and study the transfer between these tasks. Specifically, for each task, we split its data into four environments: $E_1^{\text{train}}, E_2^{\text{train}}, E^{\text{val}}, E^{\text{test}}$, where spurious correlations vary across the two training environments $E_1^{\text{train}}, E_2^{\text{train}}$. For the source task S, the model can access both of its training environments $E_1^{\text{train}}(S), E_2^{\text{train}}(S)$. For the target task T, the model only has access to one training environment $E_1^{\text{train}}(T)$. We note that the validation set $E^{\text{val}}(T)$ plays an important role in early-stopping and hyper-parameter tuning, especially when the distribution of the data is different between training and testing (Gulrajani & Lopez-Paz, 2020). In this work, since we don't have access to multiple training environments on the target task, we assume that the validation data E^{val} follows the same distribution as the training data E_1^{train}. Table 1 summarizes the level of the spurious correlations for different tasks. Additional details can be found in Appendix C.1.

Natural environments We also consider a practical setting where environments are directly defined by a given attribute of the input, and our goal is to reduce model biases from other latent attributes. We study CelebA (Liu et al., 2015a) where each input (an image of a human face) is annotated with 40 binary attributes. The source task is to predict

Table 2: Target task accuracy of different methods. All methods are tuned based on a held-out validation set that follows from the same distribution as the target training data. Bottom right: standard deviation across 5 runs. Upper right: source task testing performance (if applicable).

	SOURCE	TARGET	ERM	REUSE$_{PI}$	FINETUNE$_{PI}$	MULTITASK	TOFU	ORACLE
MNIST	ODD	EVEN	12.3$\pm$0.6	$14.4^{(70.9)}_{\pm 1.0}$	$11.2^{(70.1)}_{\pm 2.1}$	$11.6^{(69.6)}_{\pm 0.6}$	**69.1**$\pm$1.6	68.7$\pm$0.9
	EVEN	ODD	9.7$\pm$0.6	$19.2^{(71.1)}_{\pm 2.3}$	$11.5^{(71.1)}_{\pm 1.2}$	$10.1^{(70.0)}_{\pm 0.7}$	**66.8**$\pm$0.8	67.8$\pm$0.5
BEER REVIEW	LOOK	AROMA	55.5$\pm$1.7	$31.9^{(70.1)}_{\pm 1.0}$	$53.7^{(70.1)}_{\pm 1.4}$	$54.1^{(76.0)}_{\pm 2.2}$	**75.9**$\pm$1.4	77.3$\pm$1.3
	LOOK	PALATE	46.9$\pm$0.3	$22.8^{(70.0)}_{\pm 1.9}$	$49.3^{(73.2)}_{\pm 2.1}$	$52.8^{(73.3)}_{\pm 2.9}$	**73.8**$\pm$0.7	74.0$\pm$1.2
	AROMA	LOOK	63.9$\pm$0.6	$40.1^{(68.6)}_{\pm 3.1}$	$65.2^{(66.4)}_{\pm 1.8}$	$64.0^{(71.5)}_{\pm 0.6}$	**80.9**$\pm$0.5	80.1$\pm$0.6
	AROMA	PALATE	46.9$\pm$0.3	$14.0^{(68.3)}_{\pm 2.4}$	$47.9^{(63.2)}_{\pm 3.3}$	$50.0^{(71.2)}_{\pm 1.4}$	**73.5**$\pm$1.1	74.0$\pm$1.2
	PALATE	LOOK	63.9$\pm$0.6	$40.4^{(57.2)}_{\pm 2.8}$	$64.3^{(60.1)}_{\pm 2.7}$	$63.1^{(75.9)}_{\pm 1.0}$	**81.0**$\pm$1.0	80.1$\pm$0.6
	PALATE	AROMA	55.5$\pm$1.7	$23.1^{(59.2)}_{\pm 3.3}$	$54.5^{(58.7)}_{\pm 1.2}$	$56.5^{(73.3)}_{\pm 1.3}$	**76.9**$\pm$1.5	77.3$\pm$1.3
ASK.	PENE	INCI.	79.3$\pm$1.3	$71.7^{(72.7)}_{\pm 0.5}$	$79.3^{(71.2)}_{\pm 0.8}$	$71.1^{(73.5)}_{\pm 1.4}$	**83.2**$\pm$1.8	84.8$\pm$1.2
	INCI.	PENE.	71.6$\pm$1.8	$64.1^{(83.4)}_{\pm 1.5}$	$72.0^{(83.4)}_{\pm 3.1}$	$61.9^{(82.4)}_{\pm 0.7}$	**78.1**$\pm$1.4	78.3$\pm$0.9
BIRD	WATER	SEA	81.8$\pm$4.3	$87.8^{(99.5)}_{\pm 1.1}$	$82.0^{(99.5)}_{\pm 4.0}$	$88.0^{(99.5)}_{\pm 0.9}$	**93.1**$\pm$0.4	93.7$\pm$0.7
	SEA	WATER	75.1$\pm$6.3	$94.6^{(93.3)}_{\pm 1.6}$	$78.2^{(93.1)}_{\pm 8.1}$	$93.5^{(92.7)}_{\pm 1.9}$	**99.0**$\pm$0.4	98.9$\pm$0.5
	Average		55.2	43.7	55.8	56.4	**79.3**	79.6

the `Eyeglasses` attribute and the target task is to predict the `BlondHair` attribute. We use the `Young` attribute to define two environments: $E_1 = \{\texttt{Young} = 0\}$ and $E_2 = \{\texttt{Young} = 1\}$. In the source task, both environments are available. In the target task, we only have access to environment E_1 during training and validation. At test time, we evaluate the robustness of our target classifier against other latent attributes. Specifically, for each unknown attribute such as `Male`, we partition the testing data into four groups: $\{\texttt{Male} = 1, \texttt{BlondHair} = 0\}$, $\{\texttt{Male} = 0, \texttt{BlondHair} = 0\}$, $\{\texttt{Male} = 1, \texttt{BlondHair} = 1\}$, $\{\texttt{Male} = 0, \texttt{BlondHair} = 1\}$. Following (Sagawa et al., 2019), We report the worst-group accuracy and the average-group accuracy.

4.2. Baselines

We compare our algorithm against the following baselines. For fair comparison, all methods share the same representation backbone and hyper-parameter search space. Implementation details are available at Appendix C.2.

ERM baseline We learn a classifier on the target task from scratch by minimizing the average loss across all examples. Note that this classifier is independent of the source task. Its performance reflects the deviation between the training distribution and the testing distribution of the target task.

Transfer methods Since the source task contains multiple environments, we can learn a stable model on the source task and transfer it to the target task. We use four algorithms to learn the source task: DANN (Ganin et al., 2016), C-DANN (Li et al., 2018b), MMD (Li et al., 2018a), PI (Bao et al., 2021). We consider three standard methods for transferring the source knowledge:

REUSE: We directly transfer the feature extractor of the source model to the target task. The feature extractor is fixed when learning the target classifier.

FINETUNE: We update the feature extractor when training the target classifier. (Shafahi et al., 2020) has shown that FINETUNE may improve adversarial robustness of the target task.

MULTITASK: We adopt the standard multi-task learning approach (Caruana, 1997) where the source model and the target model share the same feature extractor and are jointly trained together.

Automatic de-biasing methods For the target task, we can also apply de-biasing approaches that do not require environments. We consider the following baselines:

EIIL (Creager et al., 2021): Based on a pre-trained ERM classifier's prediction logits, we infer an environment assignment that maximally violates the invariant learning principle (Arjovsky et al., 2019). We then apply group DRO to

Table 3: Worst-group and average-group accuracy on CelebA. The source task is to predict `Eyeglasses` and the target task is to predict `BlondHair`. We use the attribute `Young` to define two environments: $E_1 = \{\texttt{Young} = 0\}$, $E_2 = \{\texttt{Young} = 1\}$. Both environments are available in the source task. In the target task, we only have access to E_1 during training and validation.. We show the results for the first 3 attributes (alphabetical order). The right-most Average* column is computed based on the performance across all 38 attributes. See Appendix F for full results.

	METHOD	ArchedEyebrows		Attractive		BagsUnderEyes		Average*	
		Worst	Average	Worst	Average	Worst	Average	Worst	Average
	ERM	75.43	88.52	75.00	88.94	70.91	87.09	61.01	85.07
TRANSFER	REUSE$_{\text{PI}}$	53.71	64.05	52.13	64.85	52.50	66.88	47.58	64.14
	REUSE$_{\text{DANN}}$	59.56	72.44	62.03	72.26	64.58	73.83	55.27	72.31
	REUSE$_{\text{C-DANN}}$	56.02	67.07	57.78	67.90	57.50	68.33	53.22	68.56
	REUSE$_{\text{MMD}}$	48.91	59.80	48.46	61.51	58.74	63.11	50.61	61.27
	FINETUNE$_{\text{PI}}$	71.86	87.02	72.73	87.34	62.50	84.10	63.07	85.27
	FINETUNE$_{\text{DANN}}$	65.38	83.89	63.35	84.98	56.86	81.34	50.60	80.49
	FINETUNE$_{\text{C-DANN}}$	73.85	88.90	75.61	89.39	75.86	88.14	62.03	85.57
	FINETUNE$_{\text{MMD}}$	76.07	88.80	74.33	89.74	78.57	88.61	66.80	86.81
	MULTITASK	69.66	86.91	72.73	87.44	70.00	85.21	64.37	85.21
AUTO-DEBIAS	EIIL	64.71	85.12	64.43	85.96	66.67	83.90	57.62	83.22
	GEORGE	74.73	87.89	73.66	87.70	77.78	87.97	63.34	85.04
	LFF	45.41	60.23	47.67	60.16	42.59	60.72	42.52	62.04
	M-ADA	64.61	83.33	67.33	83.59	70.34	85.34	54.55	80.77
	DG-MMLD	69.51	87.38	68.42	87.50	63.41	84.78	55.69	83.51
	TOFU	**85.66**	**91.47**	**88.30**	**92.76**	**90.38**	**92.41**	**84.86**	**91.71**

minimize the worst-case loss over all inferred environments.

GEORGE (Sohoni et al., 2020): We use the feature representation of a pre-trained ERM classifier to cluster the training data. We then apply group DRO to minimize the worst-case loss over all clusters.

LFF (Nam et al., 2020): We train a biased classifier together with a de-biased classifier. The biased classifier amplifies its bias by minimizing the generalized cross entropy loss. The de-biased classifier then up-weights examples that are mis-labeled by the biased classifier during training.

M-ADA (Qiao et al., 2020): We use a Wasserstein auto-encoder to generate adversarial examples. The de-biased classifier is trained on both the original examples and the generated examples.

DG-MMLD (Matsuura & Harada, 2020): We iteratively divide target examples into latent domains via clustering, and train the domain-invariant feature extractor via adversarial learning.

ORACLE For synthetic environments, we can use the spurious feature to define groups and train an oracle model. For example, in task SEABIRD, this oracle model will minimize the worst-case risk over the following four groups: {`seabird` in `water`}, {`seabird` in `land`} {`landbird` in `water`}, {`landbird` in `land`}. This oracle model helps us analyze the performance of our proposed algorithm separately from the inherent limitations (such as model capacity and data size).

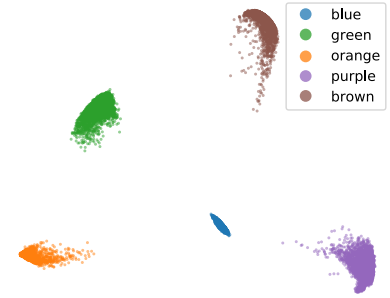

Figure 3: PCA visualization of the unstable feature representation $f_{\mathcal{Z}}$ for examples in MNIST EVEN. $f_{\mathcal{Z}}$ is trained on MNIST ODD. TOFU identifies the hidden spurious color feature by contrasting different source environments.

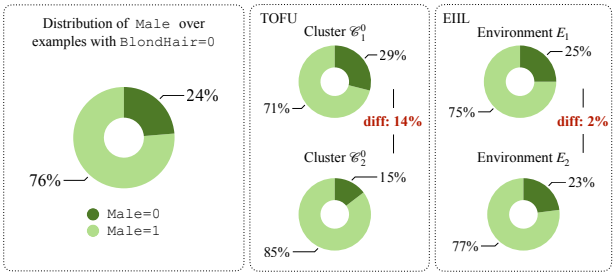

Figure 4: Visualization of the unknown attribute `Male` on CelebA. Left: distributions of `Male` in the training data. Mid: partitions learned by TOFU. Right: partitions learned by EIIL. TOFU generates partitions that are more informative of the unknown attribute (14% vs. 2%). See Appendix F for results on more attributes.

5. Results

Table 2 summarizes our results on synthetic environments. We observe that standard transfer methods fail to improve over the ERM baseline. On the other hand, TOFU consistently achieves the best performance across 12 transfer settings, outperforming the best baseline by 22.9%. While TOFU doesn't have access to the unstable features, by inferring them from the source environments, it matches the oracle performance with only 0.30% absolute difference.

Table 3 presents our results on natural environments. This task is very challenging as there are multiple latent spurious attributes in the training data. We observe that most automatic de-biasing methods underperform the ERM baseline. With the help of the source task, FINETUNE and MULTITASK achieve slightly better performance than ERM. TOFU continues to shine in this real-world setting: achieving the best worst-group and average-group performance.

Is* TOFU *able to identify the unstable features? Yes. For synthetic environments, we visualize the unstable feature representation produced by $f_{\mathcal{Z}}$ on MNIST EVEN. Figure 3 demonstrates that while $f_{\mathcal{Z}}$ only sees source examples (ODD) during training, it can distribute target examples based on their unstable color features.

For natural environments, we visualize the distribution of two latent attributes (`Male` and `ArchedEyebrows`) over the generated clusters. Figure 4 shows that the distribution gap of the unknown attribute `Male` across the generated partitions is 2% for EIIL, only marginally better than random splitting (0%). By leveraging information from the source task, TOFU learns partitions that are more informative of the unknown attribute (14%).

How do the generated clusters compare to the oracle groups? We quantitatively evaluate the generated clusters based on three metrics: *homogeneity* (each cluster contain only examples with the same unstable feature value), *completeness* (examples with the same unstable feature value belong to the same cluster), and *V-measure* (the harmonic mean of homogeneity and completeness). From Table 4, we see that TOFU is able to derive clusters that resemble the oracle groups on BEER REVIEW. In MNIST, since we generate two clusters for each label value and there are five different colors, it is impossible to recover the oracle groups. However, TOFU still achieves almost perfect completeness.

Table 4: Quantitative evaluation of the generated clusters against the ground truth unstable features. For comparison, the TRIPLET baseline (TRP) directly encourages source examples with the same label to stay close to each other in the feature space, from which we generate the clusters. For both methods, we generate two clusters for each target label value and report the average performance across all label values. We observe that the TRIPLET representation, while biased by the spurious correlations, fails to recover the ground truth unstable features for some tasks. By explicitly contrasting the source environments, TOFU derives clusters that are highly-informative of the unstable features.

SOURCE	TARGET	Homo.		Complete.		V-measure	
		TRP	TOFU	TRP	TOFU	TRP	TOFU
ODD	EVEN	0.42	**0.68**	0.58	**0.95**	0.49	**0.79**
EVEN	ODD	0.67	**0.67**	0.93	**0.99**	0.78	**0.80**
LOOK	AROMA	0.33	**0.92**	0.28	**0.92**	0.30	**0.92**
LOOK	PALATE	0.33	**0.90**	0.27	**0.89**	0.30	**0.90**
AROMA	LOOK	0.33	**1.00**	0.28	**1.00**	0.30	**1.00**
AROMA	PALATE	0.82	**1.00**	0.77	**1.00**	0.79	**1.00**
PALATE	LOOK	0.83	**0.98**	0.77	**0.98**	0.80	**0.98**
PALATE	AROMA	0.82	**0.95**	0.77	**0.95**	0.79	**0.95**

6. Conclusion

Reducing model bias is a critical problem for many machine learning applications in the real world. In this paper, we recognize that bias is a human-defined concept. Without additional knowledge, automatic de-biasing methods cannot effectively distinguish causal features from spurious features. The main departure of this paper is to identify bias by using related tasks. We demonstrate that when the source task and target task share the same set of biases, we can effectively transfer this knowledge and improve the robustness of the target model without additional human intervention. Compared with 15 baselines across 5 datasets, our approach consistently delivers significant performance gain. Quantitative and qualitative analyses confirm that our method is able to identify the hidden biases. Due to space limitations, we leave further discussions to Appendix A.

Acknowledgement

This paper is dedicated to the memory of our beloved family member Tofu, who filled our lives with so many wuffs and wuvs.

We want to thank Menghua Wu, Bracha Laufer, Adam Yala, Adam Fisch, Tal Schuster, Yujie Qian, Victor Quach and Jiang Guo for their thoughtful feedback.

Research was sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

The research, supported by the Defence Science and Technology Agency, was also funded by the Wellcome Trust (grant reference number: 222396/Z/21/Z, *Building Trustworthy Clinical AI: from algorithms to clinical deployment*).

The research was also sponsored by the MIT Jameel Clinic.

References

Ahmed, F., Bengio, Y., van Seijen, H., and Courville, A. Systematic generalisation with group invariant predictions. In *International Conference on Learning Representations*, 2020.

Ahuja, K., Shanmugam, K., Varshney, K., and Dhurandhar, A. Invariant risk minimization games. In *International Conference on Machine Learning*, pp. 145–155. PMLR, 2020.

Arjovsky, M., Bottou, L., Gulrajani, I., and Lopez-Paz, D. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Bao, Y., Deng, Z., Wang, Y., Kim, H., Armengol, V. D., Acevedo, F., Ouardaoui, N., Wang, C., Parmigiani, G., Barzilay, R., et al. Using machine learning and natural language processing to review and classify the medical literature on cancer susceptibility genes. *JCO Clinical Cancer Informatics*, 1:1–9, 2019a.

Bao, Y., Wu, M., Chang, S., and Barzilay, R. Few-shot text classification with distributional signatures. In *International Conference on Learning Representations*, 2019b.

Bao, Y., Chang, S., and Barzilay, R. Predict then interpolate: A simple algorithm to learn stable classifiers. In *International Conference on Machine Learning (ICML)*, 2021.

Beery, S., Van Horn, G., and Perona, P. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 456–473, 2018.

Belinkov, Y., Poliak, A., Shieber, S. M., Van Durme, B., and Rush, A. M. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 877–891, 2019.

Buolamwini, J. and Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pp. 77–91, 2018.

Caruana, R. Multitask learning. *Machine learning*, 28(1): 41–75, 1997.

Chang, S., Zhang, Y., Yu, M., and Jaakkola, T. S. Invariant rationalization. *arXiv preprint arXiv:2003.09772*, 2020.

Chechik, G., Sharma, V., Shalit, U., and Bengio, S. Large scale online learning of image similarity through ranking. *Journal of Machine Learning Research*, 11(3), 2010.

Clark, C., Yatskar, M., and Zettlemoyer, L. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4060–4073, 2019.

Creager, E., Jacobsen, J.-H., and Zemel, R. Environment inference for invariant learning. In *International Conference on Machine Learning*, pp. 2189–2200. PMLR, 2021.

de Vries, T., Misra, I., Wang, C., and van der Maaten, L. Does object recognition work for everyone? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 52–59, 2019.

Deng, Z., Yin, K., Bao, Y., Armengol, V. D., Wang, C., Tiwari, A., Barzilay, R., Parmigiani, G., Braun, D., and Hughes, K. S. Validation of a semiautomated natural language processing–based procedure for meta-analysis of cancer susceptibility gene penetrance. *JCO clinical cancer informatics*, 3:1–9, 2019.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Gaut, A., Sun, T., Tang, S., Huang, Y., Qian, J., ElSherief, M., Zhao, J., Mirza, D., Belding, E., Chang, K.-W., et al. Towards understanding gender bias in relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2943–2953, 2020.

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

He, H., Zha, S., and Wang, H. Unlearn dataset bias in natural language inference by fitting the residual. *EMNLP-IJCNLP 2019*, pp. 132, 2019.

Hendrycks, D., Lee, K., and Mazeika, M. Using pre-training can improve model robustness and uncertainty. In *International Conference on Machine Learning*, pp. 2712–2721. PMLR, 2019.

Hinton, G. E. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

Hu, W., Niu, G., Sato, I., and Sugiyama, M. Does distributionally robust supervised learning give robust classifiers? In *International Conference on Machine Learning*, pp. 2029–2037. PMLR, 2018.

Jia, S., Meng, T., Zhao, J., and Chang, K.-W. Mitigating gender bias amplification in distribution by posterior regularization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2936–2942, 2020.

Jin, W., Barzilay, R., and Jaakkola, T. Enforcing predictive invariance across structured biomedical domains, 2020.

Kim, Y. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1181. URL https://www.aclweb.org/anthology/D14-1181.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kiritchenko, S. and Mohammad, S. Examining gender and race bias in two hundred sentiment analysis systems. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pp. 43–53, 2018.

Krueger, D., Caballero, E., Jacobsen, J.-H., Zhang, A., Binas, J., Zhang, D., Priol, R. L., and Courville, A. Out-of-distribution generalization via risk extrapolation (rex), 2020.

Kuang, K., Xiong, R., Cui, P., Athey, S., and Li, B. Stable prediction with model misspecification and agnostic distribution shift. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4485–4492, 2020.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lei, T., Barzilay, R., and Jaakkola, T. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 107–117, 2016.

Li, H., Pan, S. J., Wang, S., and Kot, A. C. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018a.

Li, Y., Gong, M., Tian, X., Liu, T., and Tao, D. Domain generalization via conditional invariant representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018b.

Li, Z. and Xu, C. Discover the unknown biased attribute of an image classifier. In *The IEEE International Conference on Computer Vision (ICCV)*, 2021.

Liu, J., Hu, Z., Cui, P., Li, B., and Shen, Z. Heterogeneous risk minimization. *arXiv preprint arXiv:2105.03818*, 2021.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pp. 3730–3738, 2015a.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015b.

Mahabadi, R. K., Belinkov, Y., and Henderson, J. End-to-end bias mitigation by modelling biases in corpora. *arXiv preprint arXiv:1909.06321*, 2019.

Matsuura, T. and Harada, T. Domain generalization using a mixture of multiple latent domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 11749–11756, 2020.

McAuley, J., Leskovec, J., and Jurafsky, D. Learning attitudes and attributes from multi-aspect reviews. In *2012 IEEE 12th International Conference on Data Mining*, pp. 1020–1025. IEEE, 2012.

McCoy, T., Pavlick, E., and Linzen, T. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual*

Meeting of the Association for Computational Linguistics, pp. 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://www.aclweb.org/anthology/P19-1334.

Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., and Joulin, A. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Nam, J., Cha, H., Ahn, S., Lee, J., and Shin, J. Learning from failure: Training debiased classifier from biased classifier. *arXiv preprint arXiv:2007.02561*, 2020.

Oren, Y., Sagawa, S., Hashimoto, T., and Liang, P. Distributionally robust language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4218–4228, 2019.

Park, J. H., Shin, J., and Fung, P. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2799–2804, 2018.

Peters, J., Bühlmann, P., and Meinshausen, N. Causal inference by using invariant prediction: identification and confidence intervals. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, pp. 947–1012, 2016.

Qiao, F., Zhao, L., and Peng, X. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12556–12565, 2020.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Sagawa, S., Koh, P. W., Hashimoto, T. B., and Liang, P. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryxGuJrFvS.

Sakaguchi, K., Le Bras, R., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 8732–8740, 2020.

Sanh, V., Wolf, T., Belinkov, Y., and Rush, A. M. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Hf3qXoiNkR.

Schuster, T., Shah, D., Yeo, Y. J. S., Roberto Filizzola Ortiz, D., Santus, E., and Barzilay, R. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3410–3416, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1341. URL https://www.aclweb.org/anthology/D19-1341.

Shafahi, A., Saadatpanah, P., Zhu, C., Ghiasi, A., Studer, C., Jacobs, D., and Goldstein, T. Adversarially robust transfer learning. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ryebG04YvB.

Shen, Z., Cui, P., Liu, J., Zhang, T., Li, B., and Chen, Z. Stable learning via differentiated variable decorrelation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2185–2193, 2020.

Sohoni, N. S., Dunnmon, J. A., Angus, G., Gu, A., and Ré, C. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *arXiv preprint arXiv:2011.12945*, 2020.

Stacey, J., Minervini, P., Dubossarsky, H., Riedel, S., and Rocktäschel, T. Avoiding the Hypothesis-Only Bias in Natural Language Inference via Ensemble Adversarial Training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8281–8291, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.665. URL https://www.aclweb.org/anthology/2020.emnlp-main.665.

Utama, P. A., Moosavi, N. S., and Gurevych, I. Towards debiasing nlu models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 7597–7610, 2020.

Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Woodward, J. *Making things happen: A theory of causal explanation*. Oxford university press, 2005.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., et al. Analyzing learned molecular representations

for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388, 2019.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://www.aclweb.org/anthology/P19-1472.

A. Discussion

Are biases shared across real-world tasks? In this paper, we show that for tasks where the biases are shared, we can effectively transfer this knowledge to obtain a more robust model. This assumption holds in many real world applications. For example, in natural language processing, the same gender bias exist across many tasks including relation extraction (Gaut et al., 2020), semantic role labeling (Jia et al., 2020), abusive language detection (Park et al., 2018) and sentiment analysis (Kiritchenko & Mohammad, 2018). In computer vision, the same geographical bias exists across different object recognition benchmarks such as ImageNet, COCO and OpenImages (de Vries et al., 2019).

When a single source task does not describe all unwanted unstable features, we can leverage multiple source tasks and compose their individual unstable features together. We can naturally extend TOFU to accomplish this goal by learning the unstable feature representation jointly across this collection of source tasks. We focus on the basic setting in our main paper and leave this extension to Appendix E.

Our approach is not applicable in situations where the biases in the source task and the target task are completely disjoint.

What if the source task and target task are from different domains? In this paper, we focus on the setting where the source task and the target task are from the same domain. If the target task is drawn from a different domain, we can use domain-adversarial training to align the distributions of the unstable features across the source domain and the target domain (Li et al., 2018a;a). Specifically, when training the unstable feature representation f_z, we can introduce an adversarial player that tries to guess the domain label from f_z. The representation f_z is updated to fool this adversarial player in addition to minimize the triplet loss in Eq (1). We leave this extension to future work.

Can we apply domain-invariant representation learning (DIRL) directly to the source environments? Domain invariant representation learning (Ganin et al., 2016; Li et al., 2018b;a) aims to match the feature representations across domains. If we directly treat environments as domains and apply these methods, the resulting representation may still encode unstable features.

For example, in CelebA, the attribute `Male` is spuriously correlated with the target attribute `BlondHair` (Women are more likely to have blond hair than men in this dataset). Given the two environments $\{\texttt{Young} = 0\}$ and $\{\texttt{Young}{=}1\}$, DIRL learns an age-invariant representation. However, if the distribution of `Male` is the same across the two environments, DIRL will encode this attribute into the age-invariant representation (since it is helpful for predicting the target `BlondHair` attribute). In our approach, we realize that the the correlations between `Male` and `BlondHair` are different in the two environments (The elderly may have more white hair). Even though the distribution of `Male` may be the same, we can still identify this bias from the classifiers' mistakes. Empirically, Table 3 shows that while DIRL methods improve over the ERM baseline, they still perform poorly on minority groups (worst case acc 66.80% on CelebA).

What if the mistakes correspond to other factors such as label noise, distribution shifts, etc.? For ease of analysis, we do not consider label noise and distribution shift in Theorem 1. One future direction is to model bias from the information perspective (rather than looking at the linear correlations). This will enable us to relax the assumption in the analyses and we can further incorporate these different mistake factors into the modeling.

We note that we do not impose this assumption in our empirical experiments. For example, we explicitly added label noise into the MNIST data. In CELEBA, there is a distribution gap (from young people to the elderly) across the two environments. We observe that our method is able to perform robustly in situations where the assumption breaks.

Is the algorithm efficient when multiple source environments are available? Our method can be generalized efficiently to multiple environments. Given N source environments, we first note that the complexities of the target steps **T.1** and **T.2** are independent of N. For the source task, the N environment-specific classifiers (in **S.1**) can be learned jointly with multi-task learning (Caruana, 1997). This significantly reduces the inference cost at **S.2** as we only need to pass each input example through the (expensive) representation backbone for one time. In **S.3**, we sample partitions when minimizing the triplet loss, so there is no additional cost during training. In this paper, we focus on the two-environments setup for simplicity and leave this generalization to future work.

Why does the baselines perform so poorly on MNIST? We note that the representation backbone (a 2-layer CNN) on MNIST is trained from scratch while we use pre-trained representations for other datasets (see Appendix C.2). Our

hypothesis is that models are more prune to spurious correlations when trained from scratch.

B. Theoretical analysis

B.1. Partitions reveal the unstable correlation

We start by reviewing the results in (Bao et al., 2021) which shows that the generated partitions reveal the unstable correlation. We consider binary classification tasks where $\mathcal{Y} \in \{0, 1\}$. For a given input x, we use $\mathcal{C}(x)$ to represent its stable (causal) feature and $\mathcal{Z}(x)$ to represent its unstable feature. In order to ease the notation, if no confusion arises, we omit the dependency on x. We use lowercase letters c, z, y to denote the specific values of $\mathcal{C}, \mathcal{Z}, \mathcal{Y}$.

Proposition 1. *For a pair of environments E_i and E_j, assuming that the classifier f_i is able to learn the true conditional $P_i(\mathcal{Y} \mid C, \mathcal{Z})$, we can write the joint distribution P_j of E_j as the mixture of $P_j^{i\checkmark}$ and $P_j^{i\times}$:*

$$P_j(c, z, y) = \alpha_j^i P_j^{i\checkmark}(c, z, y) + (1 - \alpha_j^i) P_j^{i\times}(c, z, y),$$

where $\alpha_j^i = \sum_{c,z,y} P_j(c, z, y) \cdot P_i(y \mid c, z)$ and

$$P_j^{i\checkmark}(c, z, y) \propto P_j(c, z, y) \cdot P_i(y \mid c, z),$$
$$P_j^{i\times}(x, z, y) \propto P_j(c, z, y) \cdot P_i(1 - y \mid c, z).$$

Proof. See (Bao et al., 2021). □

Proposition 1 tells us that if f_i is powerful enough to capture the true conditional in E_i, partitioning the environment E_j is equivalent to scaling its joint distribution based on the conditional on E_i.

Now suppose that the marginal distribution of $\mathcal{Y}$ is uniform in all joint distributions, i.e., f_i performs equally well on different labels. Bao et al. (2021) shows that the unstable correlations will have different signs in the subset of correct predictions and in the subset of incorrect predictions.

Proposition 2. *Suppose $\mathcal{Z}$ is independent of $\mathcal{C}$ given $\mathcal{Y}$. For any environment pair E_i and E_j, if $\sum_y P_i(z \mid y) = \sum_y P_j(z \mid y)$ for any z, then $\mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_i) > \mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_j)$ implies*

$$\mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_j^{i\times}) < 0, \quad \textit{and} \quad \mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_i^{j\times}) > 0.$$

Proof. See (Bao et al., 2021). □

Proposition 2 implies that no matter whether the spurious correlation is positive or negative, by interpolating $P_j^{i\checkmark}, P_j^{i\times}, P_i^{j\checkmark}, P_i^{j\times}$, we can obtain an *oracle* distribution where the spurious correlation between $\mathcal{Z}$ and $\mathcal{Y}$ vanishes. Since the oracle interpolation coefficients are not available in practice, Bao et al. (2021) propose to optimize the worst-case risk across all interpolations of the partitions.

B.2. Partitions reveal the unstable feature

Proposition 2 shows that the partitions $E_j^{i\checkmark}, E_j^{i\times}, E_i^{j\checkmark}, E_i^{j\times}$ are informative of the biases. However these partitions are not transferable as they are coupled with task-specific information, i.e., the label $\mathcal{Y}$. To untangle this dependency, we look at different label values and obtain the following result.

Corollary 1. *Under the same assumption as Proposition 2, if $\mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_i) > \mathrm{Cov}(\mathcal{Z}, \mathcal{Y}; P_j) > 0$ and $\mathcal{Z}$ follows a uniform distribution within each partition, then*

$$\sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 1) > \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 1),$$
$$\sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 0) < \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 0).$$

Proof. By definition of the covariance, we have

$$\mathrm{Cov}(\mathcal{Z},\mathcal{Y}) = \sum_{z,y} zyP(\mathcal{Z}=z,\mathcal{Y}=y) - \left(\sum_{z} zP(\mathcal{Z}=z)\right)\left(\sum_{y} yP(\mathcal{Y}=y)\right)$$

Since we assume the marginal distribution of the label is uniform, we have $\sum_y yP(\mathcal{Y}=y) = 0.5$. Then we have

$$\mathrm{Cov}(\mathcal{Z},\mathcal{Y}) = \sum_{z} zP(\mathcal{Z}=z,\mathcal{Y}=1) - 0.5\sum_{z} zP(\mathcal{Z}=z).$$

Using $P(\mathcal{Z}=z) = P(\mathcal{Z}=z,\mathcal{Y}=0) + P(\mathcal{Z}=z,\mathcal{Y}=1)$, we obtain

$$\mathrm{Cov}(\mathcal{Z},\mathcal{Y}) = 0.5\sum_{z} zP(\mathcal{Z}=z,\mathcal{Y}=1) - 0.5\sum_{z} zP(\mathcal{Z}=z,\mathcal{Y}=0). \tag{2}$$

From Proposition 2, we have $\mathrm{Cov}(\mathcal{Z},\mathcal{Y};P_j^{i\times}) < 0$. Note that this implies $\mathrm{Cov}(\mathcal{Z},\mathcal{Y};P_j^{i\checkmark}) > 0$ since $\mathrm{Cov}(\mathcal{Z},\mathcal{Y};P_j) > 0$ and $P_j = \alpha_j^i P_j^{i\checkmark} + (1-\alpha_j^i)P_j^{i\times}$. Combining with Eq (2), we have

$$\begin{aligned}
\sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=1) &< \sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=0),\\
\sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=1) &> \sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=0).
\end{aligned} \tag{3}$$

Since we assume the marginal distribution of the unstable feature $\mathcal{Z}$ is uniform, we have

$$\begin{aligned}
\sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=1) + \sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=0) &= \sum_{z} zP_j^{i\times}(\mathcal{Z}=z) = 0.5,\\
\sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=1) + \sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=0) &= \sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z) = 0.5.
\end{aligned} \tag{4}$$

Plugging Eq (4) into Eq (4), we have

$$\begin{aligned}
\sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=1) &< 0.25 < \sum_{z} zP_j^{i\times}(\mathcal{Z}=z,\mathcal{Y}=0),\\
\sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=1) &> 0.25 > \sum_{z} zP_j^{i\checkmark}(\mathcal{Z}=z,\mathcal{Y}=0).
\end{aligned}$$

Combining the two inequalities finishes the proof. □

Corollary 1 shows that if we look at examples within the same label value, then expectation of the unstable feature $\mathcal{Z}$ within the set of correct predictions will diverge from the one within the set of incorrect predictions. In order to learn a metric space that corresponds to the values of $\mathcal{Z}$, we sample different batches from the partitions and prove the following theorem.

Theorem 1. *(Full version) Suppose $\mathcal{Z}$ is independent of $\mathcal{C}$ given $\mathcal{Y}$. We assume that $\mathcal{Y}$ and $\mathcal{Z}$ both follow a uniform distribution within each partition.*

Consider examples in E_j with label value y. Let $X_1^{\checkmark}, X_2^{\checkmark}$ denote two batches of examples that f_i predicted correctly, and let $X_3^{\times}$ denote a batch of incorrect predictions. If $\mathrm{Cov}(\mathcal{Z},\mathcal{Y};P_i) > \mathrm{Cov}(\mathcal{Z},\mathcal{Y};P_j) > 0$, we have

$$\|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_2^{\checkmark})\|_2 < \|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_3^{\times})\|_2$$

almost surely for large enough batch size.

Proof. Without loss of generality, we consider $y=0$. Let n denote the batch size of $X_1^{\checkmark}$, $X_2^{\checkmark}$ and $X_3^{\checkmark}$. By the law of large numbers, we have

$$\overline{\mathcal{Z}}(X_1^{\checkmark}), \overline{\mathcal{Z}}(X_2^{\checkmark}) \xrightarrow{\text{a.s.}} \mathbb{E}_{P_j^{i\checkmark}(\mathcal{Z}|\mathcal{Y})}\left[\mathcal{Z} \mid \mathcal{Y}=0\right] \quad \text{and} \quad \overline{\mathcal{Z}}(X_3^{\times}) \xrightarrow{\text{a.s.}} \mathbb{E}_{P_j^{i\times}(\mathcal{Z}|\mathcal{Y})}\left[\mathcal{Z} \mid \mathcal{Y}=0\right],$$

as $n \to \infty$. Note that Corollary 1 tells us

$$\mathbb{E}_{P_j^{i\times}(\mathcal{Z}|\mathcal{Y})}\left[\mathcal{Z} \mid \mathcal{Y}=0\right] < \mathbb{E}_{P_j^{i\checkmark}(\mathcal{Z}|\mathcal{Y})}\left[\mathcal{Z} \mid \mathcal{Y}=0\right].$$

Thus we have

$$\|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_2^{\checkmark})\|_2 < \|\overline{\mathcal{Z}}(X_1^{\checkmark}) - \overline{\mathcal{Z}}(X_3^{\times})\|_2$$

almost surely as $n \to \infty$. □

We note that while we focus our theoretical analysis on binary tasks, empirically, our method is able to correctly identify the hidden bias for multi-dimensional unstable features and multi-dimensional label values.

C. Experimental setup

C.1. Datasets and models

C.1.1. MNIST

Data We extend Arjovsky et al. (2019)'s approach for generating spurious correlations and define two *multi-class* classification tasks: EVEN (5-way classification among digits `0, 2, 4, 6, 8`) and ODD (5-way classification among digits `1, 3, 5, 7, 9`). For each image, we first map its numeric digit value y^{digit} into its class id within the task: $y^{\text{causal}} = \lfloor y^{\text{digit}}/2 \rfloor$. This class id serves as the causal feature for the given task. We then sample the observed label y, which equals to y^{causal} with probability 0.75 and a uniformly random other label value with the remaining probability. With this noisy label, we now sample the spurious color feature: the color value equals y with η probability and a uniformly other value with the remaining probability. We note that since there are five different digits for each task, we have five different colors. Finally, we color the image according to the generated color value. For the training environments, we set η to 0.8 in E_1^{train} and 0.9 in E_2^{train}. We set $\eta = 0.1$ in the testing environment E^{test}.

We use the official train-test split of MNIST. Training environments are constructed from training split, with 7370 examples per environment for EVEN and 7625 examples per environment for ODD. Validation data and testing data is constructed based on the testing split. For EVEN, both validation data and testing data have 1230 examples. For ODD, the number is 1267. Following Arjovsky et al. (2019), We convert each grey scale image into a $5 \times 28 \times 28$ tensor, where the first dimension corresponds to the spurious color feature.

Representation backbone We follow the architecture from PyTorch's MNIST example[2]. Specifically, each input image is passed to a CNN with 2 convolution layers followed by 2 fully connected layers.

License The dataset is freely available at `http://yann.lecun.com/exdb/mnist/`.

C.1.2. BEER REVIEW

Data We consider the transfer among three *binary* aspect-level sentiment classification tasks: LOOK, AROMA and PALATE (Lei et al., 2016). For each review, we follow Bao et al. (2021) and append a pseudo token (`art_pos` or `art_neg`) based on the the sentiment of the given aspect (`pos` or `neg`). The probability that this pseudo token agrees with the sentiment label is 0.8 in E_1^{train} and 0.9 in E_2^{train}. In the testing environment, this probability reduces to 0.1. Unlike MNIST, there is no label noise added to the data.

We use the script created by Bao et al. (2021) to generate spurious features for each aspect. Specifically, for each aspect, we randomly sample training/validation/testing data from the dataset. Since our focus in this paper is to measure whether the algorithm is able to remove biases (rather than label imbalance), we maintain the marginal distribution of the label to be uniform. Each training environment contains 4998 examples. The validation data contains 4998 examples and the testing data contains 5000 examples. The vocabulary sizes for the three aspects (look, aroma, palate) are: 10218, 10154 and 10086.

Representation backbone We use a 1D CNN (Kim, 2014), with filter size $3, 4, 5$, to obtain the feature representation. Specifically, each input is first encoded by pre-trained FastText embeddings (Mikolov et al., 2018). Then it is passed into a convolution layer followed by max pooling and ReLU activation.

[2]https://github.com/pytorch/examples/blob/master/mnist/main.py

License This dataset was originally downloaded from `https://snap.stanford.edu/data/web-BeerAdvocate.html`. As per request from BeerAdvocate the data is no longer publicly available.

C.1.3. ASK2ME

Data ASK2ME (Bao et al., 2019a) is a text classification dataset where the inputs are paper abstracts from PubMed. We study the transfer between two *binary* classification tasks: PENETRANCE (identifying whether the abstract is informative about the risk of cancer for gene mutation carriers) and INCIDENCE (identifying whether the abstract is informative about proportion of gene mutation carriers in the general population). By definition, both tasks are causally-independent of the diseases that have been studied in the abstract. However, due to the bias in the data collection process, Deng et al. (2019) found that the performance varies (by 12%) when we evaluate based on different cancers. To assess whether we can remove such bias, we define two training environments for each task based on the correlations between the task label and the `breast_cancer` attribute (indicating the presence of breast cancer in the abstract). Script for generating the environments is available in the supplemental materials. Note that the model doesn't have access to the `breast_cancer` attribute during training.

Following Sagawa et al. (2019), we evaluate the performance on a balanced test environment where there is no spurious correlation between `breast_cancer` and the task label. This helps us understand the overall generalization performance across different input distributions.

We randomly split the data and use 50% for PENETRANCE and 50% for INCIDENCE. For PENETRANCE, there are 948 examples in E_1^{train} and E^{val}, 816 examples in E_2^{train} and 268 examples in E^{test}. For INCIDENCE, there are 879 examples in E_1^{train} and E^{val}, 773 examples in E_2^{train} and 548 examples in E^{test}. The processed data will be publicly available.

Representationi backbone The model architecture is the same as the one for Beer review.

License MIT License.

C.1.4. WATERBIRD

Data Waterbird is an image classification dataset where each image is labeled based on its bird class (Welinder et al., 2010) and the background attribute (`water` vs. `land`). Following Sagawa et al. (2019), we group different bird classes together and consider two *binary* classification tasks: SEABIRD (classifying 36 seabirds against 36 landbirds) and WATERFOWL (classifying 9 waterfowl against 9 *different* landbirds). Similar to ASK2ME, we define two training environments for each task based on the correlations between the task label and the `background` attribute. Script for generating the environments is available in the supplemental materials. At test time, we measure the generalization performance on a balanced test environment.

Following Liu et al. (2015b), we group different classes of birds together to form binary classification tasks.

In WATERFOWL, the task is to identify 9 different waterfowls (Red breasted Merganser, Pigeon Guillemot, Horned Grebe, Eared Grebe, Mallard, Western Grebe, Gadwall, Hooded Merganser, Pied billed Grebe) against 9 different landbirds (Mourning Warbler, Whip poor Will, Brewer Blackbird, Tennessee Warbler, Winter Wren, Loggerhead Shrike, Blue winged Warbler, White crowned Sparrow, Yellow bellied Flycatche). The training environment E_1^{train} contains 298 examples and the training environment E_2^{train} contains 250 examples. The validation set has 300 examples and the test set has 216 examples.

In SEABIRD, the task is to identify *36 different seabirds* (Heermann Gull, Red legged Kittiwake, Rhinoceros Auklet, White Pelican, Parakeet Auklet, Western Gull, Slaty backed Gull, Frigatebird, Western Meadowlark, Long tailed Jaeger, Red faced Cormorant, Pelagic Cormorant, Brandt Cormorant, Black footed Albatross, Western Wood Pewee, Forsters Tern, Glaucous winged Gull, Pomarine Jaeger, Sooty Albatross, Artic Tern, California Gull, Horned Puffin, Crested Auklet, Elegant Tern, Common Tern, Least Auklet, Northern Fulmar, Ring billed Gull, Ivory Gull, Laysan Albatross, Least Tern, Black Tern, Caspian Tern, Brown Pelican, Herring Gull, Eastern Towhee) against *36 different landbirds* (Prairie Warbler, Ringed Kingfisher, Warbling Vireo, American Goldfinch, Black and white Warbler, Marsh Wren, Acadian Flycatcher, Philadelphia Vireo, Henslow Sparrow, Scissor tailed Flycatcher, Evening Grosbeak, Green Violetear, Indigo Bunting, Gray Catbird, House Sparrow, Black capped Vireo, Yellow Warbler, Common Raven, Pine Warbler, Vesper Sparrow, Pileated Woodpecker, Bohemian Waxwing, Bronzed Cowbird, American Three toed Woodpecker, Northern Waterthrush, White breasted Kingfisher, Olive sided Flycatcher, Song Sparrow, Le Conte Sparrow, Geococcyx, Blue Grosbeak, Red cockaded

Table 5: Data statistics of CelebA. The model has access to both E_1 and E_2 on the source task. For the target task, only E_1 is available during training and validation.

	source task `Eyeglasses`		target task `BlondHair`	
	$E_1 : \{\texttt{Young=0}\}$	$E_2 : \{\texttt{Young=1}\}$	$E_1 : \{\texttt{Young=0}\}$	$E_2 : \{\texttt{Young=1}\}$
Train	17955	63430	17973	63412
Val	2494	7442	2453	7480
Test	2452	7597	2444	7537

Woodpecker, Green tailed Towhee, Sayornis, Field Sparrow, Worm eating Warbler). The training environment E_1^{train} contains 1176 examples and the training environment E_2^{train} contains 998 examples. The validation set has 1179 examples and the test set has 844 examples.

Representation backbone We use the Pytorch torchvision implementation of the ResNet50 model, starting from pretrained weights. We re-initalize the final layer to predict the label.

License This dataset is publicly available at `https://nlp.stanford.edu/data/dro/waterbird_complete95_forest2water2.tar.gz`

C.1.5. CELEBA

Data CelebA (Liu et al., 2015a) is an image classification dataset where each image is annotated with 40 binary attributes. We consider `Eyeglasses` as the source task and `BlondHair` as the target task. We split the official train / val / test set into two parts (uniformly at random) for each task. We use the attribute `Young` to create two environments: $E_1 = \{\texttt{Young} = 0\}$, $E_2 = \{\texttt{Young} = 1\}$. For the target task, the model only has access to E_1 during training and validation. Table 5 summarizes the data statistics.

License The CelebA dataset is available for non-commercial research purposes only. It is publicly available at `https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`.

Representation backbone We use the Pytorch torchvision implementation of the ResNet50 model, starting from pretrained weights. We re-initalize the final layer to predict the label.

C.2. Implementation details

For all methods: We use batch size 50 and evaluate the validation performance every 100 batch. We apply early stopping once the validation performance hasn't improved in the past 20 evaluations. We use Adam (Kingma & Ba, 2014) to optimize the parameters and tune the learning rate $\in \{10^{-3}, 10^{-4}\}$. For simplicity, we train all methods without data augmentation. Following Sagawa et al. (2019), we apply strong regularizations to avoid over-fitting. Specifically, we tune the dropout rate $\in \{0.1, 0.3, 0.5\}$ for text classification datasets (Beer review and ASK2ME) and tune the weight decay parameters $\in \{10^{-1}, 10^{-2}, 10^{-3}\}$ for image datasets (MNIST, Waterbird and CelebA).

DANN, C-DANN For the domain adversarial network, we use a MLP with 2 hidden ReLU layer with 300 neurons for each layer. The representation backbone is updated via a gradient reversal layer. We tune the weight of the adversarial loss $\in \{0.01, 0.1, 1\}$.

MMD We match the mean and covariance of the features across the two source environments. We use the implementation from `https://github.com/facebookresearch/DomainBed/blob/main/domainbed/algorithms.py`. We tune the weight of the MMD loss $\in \{0.01, 0.1, 1\}$.

MULTITASK For the source task, we first partition the source data into subsets with opposite spurious correlations (Bao et al., 2021). During multi-task training, we minimize the worst-case risk over all these subsets for the source task and minimize the average empirical risk for the target task. MULTITASK is more flexible than REUSE since we tune feature

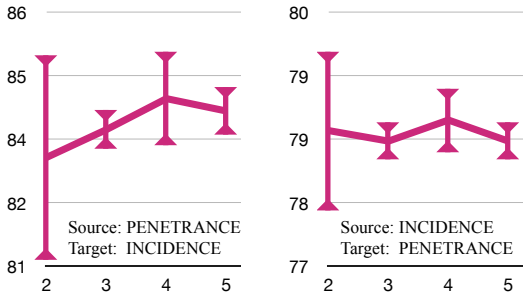

Figure 5: Accuracy of TOFU on ASK2ME as we vary the number of clusters n_c generated for each label value. Empirically, we see that while having more clusters doesn't improve the performance, it helps reduce the variance.

extractor to fit the target data. Compared to FINETUNE, MULTITASK is more constrained as the source model prevents over-utilization of unstable features during joint training.

Ours We fix $\delta = 0.3$ in all our experiments. Based on our preliminary experiments (Figure 5), we fix the number of clusters to be 2 for all our experiments in Table 2 and Table 3. For the target classifier, we directly optimize the $\min - \max$ objective. Specifically, at each step, we sample a batch of example from each group, and minimize the worst-group loss. We found the training process to be pretty stable when using the Adam optimizer.

Validation criteria For ERM, REUSE, FINETUNE and MULTITASK, since we don't have any additional information (such as environments) for the target data, we apply early stopping and hyper-parameter selection based on the average accuracy on the validation data.

For TOFU, since we have already learned an unstable feature representation $f_{\mathcal{Z}}$ on the source task, we can also use it to cluster the validation data into groups where the unstable features within each group are different. We measure the worst-group accuracy and use it as our validation criteria.

For ORACLE, as we assume access to the oracle unstable features for the target data, we can use them to define groups on the validation data as well. We use the worst-group accuracy as our validation criteria.

We also note that when we transfer from LOOK to AROMA in Table 2, both TOFU and ORACLE are able to achieve 75 accuracy on E^{test}. This number is higher than the performance of training on AROMA with two data environments (68 accuracy in Table 2). This result makes sense since in the latter case, we only have in-domain validation set and we use the average accuracy as our hyper-parameter selection metric. However, in both TOFU and ORACLE, we create (either automatically or manually) groups over the validation data and measure the worst-group performance. This ensures that the chosen model will not over-fit to the unstable correlations.

Computational resources: We use our internal clusters (24 NVIDIA RTX A6000 and 16 Tesla V100-PCIE-32GB) for the experiments. It took around a week to generate all the results in Table 2 and Table 3.

D. Additional analysis

Why do the baselines behave so differently across different datasets? As Bao et al. (2019b) pointed out, the transferability of the low-level features is very different in image classification and in text classification. For example, the keywords for identifying the sentiment of LOOK are very different from the ones for PALATE. Thus, fine-tuning the feature extractor is crucial. This explains why REUSE underperforms other baselines on text data. Conversely, in image classification, the low-level patterns (such as edges) are more transferable across tasks. Directly reusing the feature extractor helps improve model stability against spurious correlations. Finally, we note that since TOFU transfers the unstable features instead of the task-specific causal features, it performs robustly across all the settings.

How many clusters to generate? We study the effect of the number of clusters on ASK2ME. Figure 5 shows that while generating more clusters in the unstable feature space $f_{\mathcal{Z}}$ reduces the variance, it doesn't improve the performance by much. This is not very surprising as the training data is primarily biased by a single `breast_cancer` attribute. We expect that

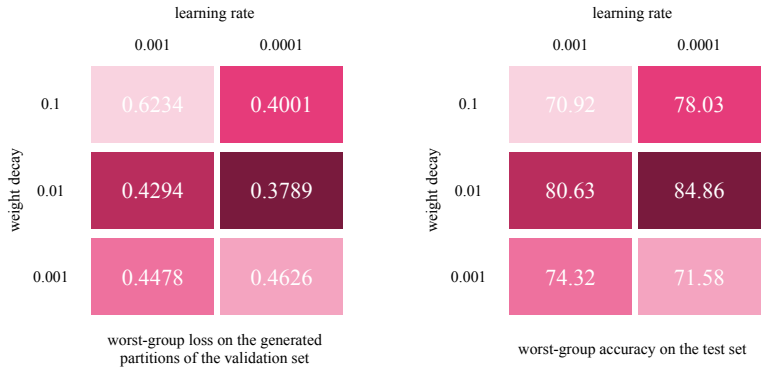

Figure 6: Hyper-parameter selection for TOFU on CelebA (averaged across 5 runs). We use our learned unstable feature representation $f_{\mathcal{Z}}$ to partition the validation set and use the worst-group validation loss as our hyper-parameter selection criteria. Empirically, we observe that this criteria correlates well with the model robustness on the testing data.

Table 6: Ablation study on MNIST and BEER REVIEW.

	SOURCE	TARGET	FINETUNE$_{\text{PI}}$	ABLATION	TOFU
MNIST	ODD	EVEN	11.2	19.2	**69.1**
	EVEN	ODD	11.5	18.7	**66.8**
BEER REVIEW	LOOK	AROMA	53.7	71.4	**75.9**
	LOOK	PALATE	49.3	64.6	**73.8**
	AROMA	LOOK	65.2	69.6	**80.9**
	AROMA	PALATE	47.9	50.3	**73.5**
	PALATE	LOOK	64.3	70.1	**81.0**
	PALATE	AROMA	54.5	57.9	**76.9**

having more clusters will be beneficial for tasks with more sophisticated underlying biases.

How do we select the hyper-parameter for* TOFU*? We cluster the validation data based on the learned unstable feature representation $f_{\mathcal{Z}}$ and use the worst-group loss as our early stopping and hyper-parameter selection criteria. Figure 6 shows our hyper-parameter search space. We observe that our validation criteria correlates well with the robustness of the model on the testing data.

Ablation study The procedures in TOFU are interconnected. For example, we cannot apply group DRO (**T.2**) without clustering the target examples (**T.1**). Nevertheless, we can empirically verify that TOFU provides cleaner separation in the unstable feature space than a naive variant.

In Theorem 1, we show that it is necessary to compare example pairs with the same label value, as opposed to contrasting all example pairs. Otherwise, the unstable features will be coupled with the *task-specific* information. To empirically support our design choice, we consider a variant of TOFU that minimizes the hinge loss over all example pairs in **S.3**:

S.3 (ABLATION) Learn a feature representation $f_{\mathcal{Z}}$ by minimizing Eq (1) across all pairs of environments E_i, E_j:

$$f_{\mathcal{Z}} = \arg\min \sum_{E_i \neq E_j} \mathbb{E}_{X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}} \left[\mathcal{L}_{\mathcal{Z}}(X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}) \right],$$

where batches $X_1^{\checkmark}, X_2^{\checkmark}$ are sampled uniformly from $E_j^{i\checkmark}$ and batch $X_3^{\times}$ is sampled uniformly from $E_j^{i\times}$.

We can think of ABLATION as directly transferring the partitions from the source task to the target task. Table 6 presents the results on MNIST and BEER REVIEW. We observe that while ABLATION slightly improves over the fine-tuning baseline, it

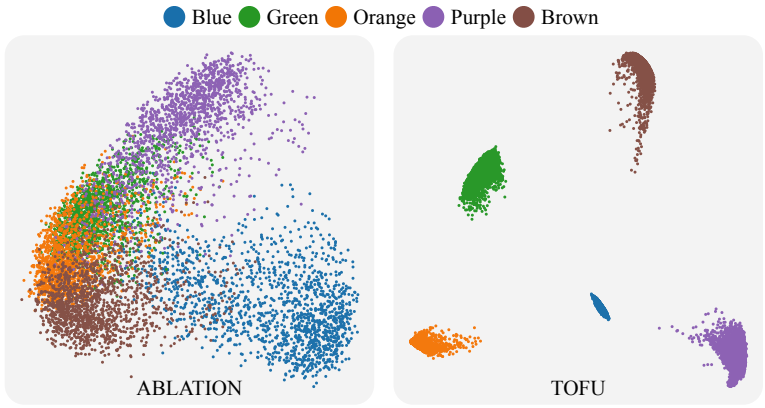

Figure 7: PCA visualizations of the feature representation $f_{\mathcal{Z}}$ for examples in MNIST EVEN (best viewed in color). $f_{\mathcal{Z}}$ is trained on MNIST ODD. Compared with ABLATION (left), TOFU perfectly separates target examples based on their spurious features (color).

Table 7: Illustration of the tasks on MNIST for multiple source tasks experiments. In the source tasks (S_1, S_2, S_3), we want to classify two digits where the label is spuriously correlated with a color pair (red-blue, red-green, blue-green). In the target task T, the goal is to learn a color-invariant model by using only one *biased* environment E_1^{train}.

Tasks	Labels	E_1^{train}	E_2^{train}	E^{test}
S_1	0 vs. 1	0000000000 1111111111	0000000000 1111111111	0000000000 1111111111
S_2	2 vs. 3	2222222222 3333333333	2222222222 3333333333	2222222222 3333333333
S_3	4 vs. 5	4444444444 5555555555	4444444444 5555555555	4444444444 5555555555
T	6 vs. 7 vs. 8	6666666666 7777777777 8888888888	NA	6666666666 7777777777 8888888888

significantly underperforms TOFU across all transfer settings. Figure 7 further confirms that the feature space learned by ABLATION are not representative of the unstable color feature. We will include this ablation analysis in our updated version.

E. Multiple source tasks

One major limitation of our work is that the source task and the target task need to share the same unstable features. While a single source task may not describe all unwanted unstable features, we can leverage multiple source tasks and combine their individual unstable features together.

Extending TOFU to multiple source tasks We can naturally extend our algorithm by inferring a *joint* unstable feature space across all source tasks.

MS.1 For each source task S and for each source environment ${}^{S}E_i$, train an environment-specific classifier ${}^{S}f_i$.

MS.2 For each source task S and for each pair of environments ${}^{S}E_i$ and ${}^{S}E_j$, use classifier ${}^{S}f_i$ to partition ${}^{S}E_j$ into two sets: ${}^{S}E_j^{i\checkmark}$ and ${}^{S}E_j^{i\times}$, where ${}^{S}E_j^{i\checkmark}$ contains examples that ${}^{S}f_i$ predicted correctly and ${}^{S}E_j^{i\times}$ contains those predicted incorrectly.

MS.3 Learn an unstable feature representation $f_{\mathcal{Z}}$ by minimizing Eq (1) across all source tasks S, all pairs of environments

Table 8: Target task testing accuracy of different methods on MNIST with different combinations of the source tasks (see Table 7 for an illustration of the tasks). Majority baseline is 33%. All methods are tuned based on a held-out validation set that follows from the same distribution as the target training data. Bottom right: standard deviation across 5 runs. Upper right: avg. source task testing performance (if applicable).

SOURCE	ERM	REUSE$_{\text{PI}}$	FINETUNE$_{\text{PI}}$	MULTITASK	TOFU	ORACLE
S_1	$26.8_{\pm 2.4}$	$34.7^{(72.0)}_{\pm 5.0}$	$35.1^{(71.9)}_{\pm 2.4}$	$17.7^{(69.4)}_{\pm 0.3}$	**57.3**$_{\pm 6.9}$	$72.7_{\pm 0.7}$
S_2	$26.8_{\pm 2.4}$	$34.6^{(68.0)}_{\pm 1.7}$	$31.0^{(66.7)}_{\pm 0.8}$	$14.6^{(74.5)}_{\pm 2.3}$	**57.8**$_{\pm 8.3}$	$72.7_{\pm 0.7}$
S_3	$26.8_{\pm 2.4}$	$34.1^{(70.2)}_{\pm 0.8}$	$33.6^{(66.3)}_{\pm 0.7}$	$12.9^{(71.2)}_{\pm 3.4}$	**49.8**$_{\pm 5.2}$	$72.7_{\pm 0.7}$
$S_1 + S_2$	$26.8_{\pm 2.4}$	$34.0^{(67.9)}_{\pm 13.9}$	$18.3^{(68.2)}_{\pm 3.2}$	$22.2^{(71.3)}_{\pm 3.0}$	**52.9**$_{\pm 1.0}$	$72.7_{\pm 0.7}$
$S_1 + S_3$	$26.8_{\pm 2.4}$	$49.9^{(70.3)}_{\pm 15.7}$	$48.3^{(68.7)}_{\pm 15.3}$	$20.3^{(72.3)}_{\pm 2.8}$	**53.4**$_{\pm 2.3}$	$72.7_{\pm 0.7}$
$S_2 + S_3$	$26.8_{\pm 2.4}$	$49.5^{(71.3)}_{\pm 7.5}$	$50.9^{(72.0)}_{\pm 12.0}$	$18.5^{(74.6)}_{\pm 7.5}$	**53.4**$_{\pm 4.1}$	$72.7_{\pm 0.7}$
$S_1 + S_2 + S_3$	$26.8_{\pm 2.4}$	$34.1^{(69.0)}_{\pm 16.4}$	$40.3^{(68.5)}_{\pm 26.3}$	$26.4^{(71.0)}_{\pm 1.2}$	**72.3**$_{\pm 1.5}$	$72.7_{\pm 0.7}$

${}^S E_i, {}^S E_j$ and all possible label value y:

$$f_{\mathcal{Z}} = \arg\min \sum_{S} \sum_{y, {}^S E_i \neq {}^S E_j} \mathbb{E}_{X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}} \left[\mathcal{L}_{\mathcal{Z}}(X_1^{\checkmark}, X_2^{\checkmark}, X_3^{\times}) \right],$$

where batches $X_1^{\checkmark}, X_2^{\checkmark}$ are sampled uniformly from ${}^S E_j^{i\checkmark}|_y$ and batch $X_3^{\times}$ is sampled uniformly from ${}^S E_j^{i\times}|_y$ ($\cdot|_y$ denotes the subset of $\cdot$ with label value y).

On the target task, we use this joint unstable feature representation $f_{\mathcal{Z}}$ to generate clusters as in Section 3.2. Since $f_{\mathcal{Z}}$ is trained across the source tasks, the generated clusters are informative of all unstable features that are present in these tasks. By minimizing the worst-case risks across the clusters, we obtain the final stable classifier.

Experiment setup We design controlled experiments on MNIST to study the effect of having multiple source tasks. We consider three source tasks: S_1 (`0` vs. `1`), S_2 (`2` vs `3`) and S_3 (`4` vs. `5`). For the target task T, the goal is to identify `6`, `7` and `8`.

Similar to Section 4, we first generated the observed noisy label based on the digits. We then inject spurious color features to the input images. For S_1, S_2 and S_3, the noisy labels are correlated with `red`/`blue`, `red`/`green` and `blue`/`green` respectively. For the target task T, the three noisy labels (`6`/`7`/`8`) are correlated with all three colors `red`/`blue`/`green`. Table 7 illustrate the different spurious correlations across the tasks.

Baselines Since ERM and ORACLE only depend on the target task, they are the same as we described in Section 4. For REUSE and FINETUNE, we first use multitask learning to learn a shared feature representation across all tasks. Specifically, for each source task, we first partition its data into subsets with opposite spurious correlations by contrasting its data environments E_1^{train} and E_2^{train} (Bao et al., 2021). We then train a joint model, with a different classifier head for each source task, by minimizing the worst-case risk over all these subsets for each source task. The shared feature representation is directly transferred to the target task. The baseline MULTITASK is similar to REUSE and FINETUNE. The difference is that we jointly train the target task classifier together with all source tasks' classifiers.

Results Table 8 presents our results on learning from multiple source tasks. Compared with the baselines, TOFU achieves the best performance across all 7 transfer settings.

We observe that having two tasks doesn't necessarily improve the target performance for TOFU. This result is actually not surprising. For example, let's consider having two source tasks S_1 and S_2. TOFU learns to recognize `red` vs. `blue` from S_1 and `red` vs. `green` from S_2, but TOFU doesn't know that `blue` should be separated from `green` in the unstable feature space. Therefore, we shouldn't expect to see any performance improvement when we combine S_1 and S_2. However, if we

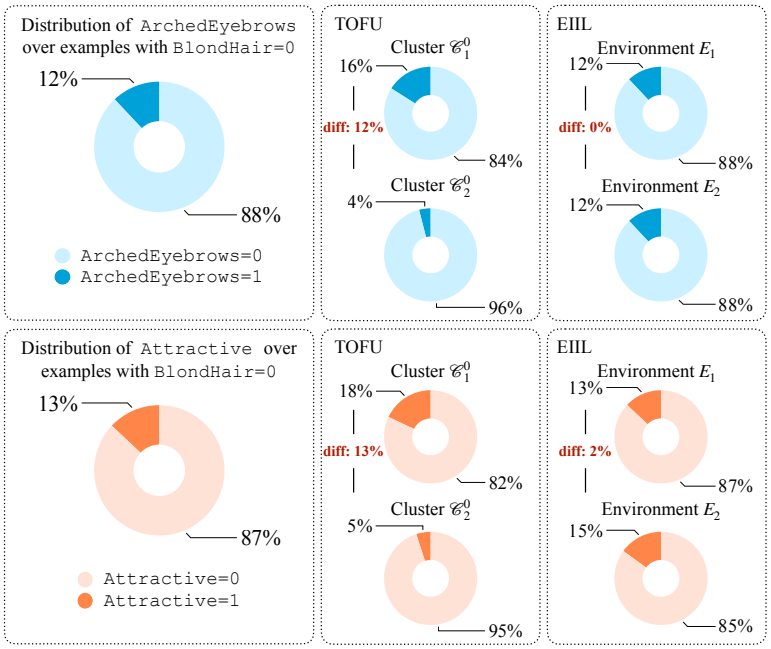

Figure 8: Visualization of the unknown attribute ArchedEyebrows and Attractive on CelebA. Left: distributions of ArchedEyebrows and Attractive in the training data. Mid: partitions learned by TOFU. Right: partitions learned by EIIL.

have one more source task S_3 which specifies the invariance between blue and green, TOFU is able to achieve the oracle performance.

For the direct transfer baselines, we see that MULTITASK simply learns to overfit the spurious correlation and performs similar to ERM. REUSE and FINETUNE generally perform better when more source tasks are available. However, their testing performance vary a lot across different runs.

F. Full results on CelebA

Table 9: Worst-group accuracy on CelebA. The source task is to predict `Eyeglasses` and the target task is to predict `BlondHair`. We use the attribute `Young` to define two environments: $E_1 = \{\texttt{Young} = 0\}$, $E_2 = \{\texttt{Young} = 1\}$. Both environments are available in the source task. In the target task, we only have access to E_1 during training and validation.

Methods	ERM	FINETUNE PI	FINETUNE DANN	FINETUNE C-DANN	FINETUNE MMD	REUSE PI	REUSE DANN	REUSE C-DANN	REUSE MMD	MULTI TASK	EIIL	GEORGE	LFF	M-ADA	DG-MMLD	TOFU
Arched_Eyebrows	75.43	71.86	65.38	73.85	76.07	53.71	59.56	56.02	48.91	69.66	64.71	74.73	45.41	64.61	69.51	85.66
Attractive	75.00	72.73	63.35	75.61	74.33	52.13	62.03	57.78	48.46	72.73	64.43	73.66	47.67	67.33	68.42	88.30
Bags_Under_Eyes	70.91	62.50	56.86	75.86	78.57	52.50	64.58	57.50	58.74	70.00	66.67	77.78	42.59	70.34	63.41	90.38
Bald	80.79	76.84	71.14	80.92	79.58	55.56	67.99	61.70	60.00	77.18	71.71	79.07	52.05	71.57	77.30	91.53
Bangs	76.06	69.33	63.59	77.11	71.76	51.15	64.67	53.42	59.29	70.22	65.29	76.74	48.04	63.54	66.88	88.70
Big_Lips	75.29	72.73	64.46	73.03	73.89	54.41	66.09	59.24	58.75	72.00	69.32	78.24	47.88	70.79	69.13	88.66
Big_Nose	80.65	71.43	68.29	79.17	76.47	54.33	63.27	58.90	51.16	76.59	71.43	78.76	48.84	70.93	68.29	88.89
Black_Hair	80.79	76.84	71.14	80.92	79.58	55.56	67.99	61.70	60.00	77.18	71.71	79.07	52.05	71.57	77.30	91.53
Blurry	68.75	62.07	58.06	64.71	65.52	52.94	67.39	56.25	28.12	56.76	50.00	75.86	29.03	34.48	48.15	80.65
Brown_Hair	60.00	25.00	16.67	33.33	50.00	50.00	33.33	50.00	57.71	66.67	0.00	57.14	51.74	60.00	66.67	80.00
Bushy_Eyebrows	80.79	76.67	66.67	80.75	50.00	55.56	67.99	61.57	52.66	77.18	50.00	50.00	51.89	50.00	50.00	91.47
Chubby	57.14	20.00	12.50	25.00	28.57	33.33	40.00	60.00	60.00	33.33	40.00	33.33	51.05	33.33	77.22	57.14
Double_Chin	64.29	50.00	70.83	80.00	64.29	50.00	60.00	60.00	41.67	50.00	55.56	50.00	51.62	54.55	30.00	87.50
Eyeglasses	60.00	68.75	53.85	75.00	58.33	15.38	6.25	15.38	0.00	60.00	42.86	76.47	22.22	69.23	40.00	73.33
Goatee	0.00	76.84	0.00	0.00	79.58	55.56	67.99	61.70	60.00	77.10	71.71	0.00	52.05	0.00	0.00	91.53
Gray_Hair	64.29	54.55	54.55	63.64	66.67	55.44	14.29	33.33	59.85	44.44	37.50	73.33	20.00	71.28	37.50	85.71
Heavy_Makeup	70.95	65.89	56.85	70.59	71.53	52.48	56.49	50.76	44.53	66.91	54.86	70.83	38.10	62.24	60.63	84.25
High_Cheekbones	65.96	62.96	58.62	72.34	75.61	47.31	55.29	50.53	45.88	66.33	54.00	68.82	37.21	51.09	62.96	86.73
Male	22.86	20.00	21.62	26.32	34.48	32.00	26.47	43.33	39.29	33.33	23.53	40.62	20.00	21.88	10.53	66.67
Mouth_Slightly_Open	75.47	73.64	68.10	77.86	79.55	45.61	64.71	57.02	57.28	76.03	67.44	78.33	48.57	66.39	76.47	91.01
Mustache	80.79	76.84	71.14	80.92	79.58	55.56	67.99	61.70	60.00	77.18	71.71	79.07	52.05	71.57	77.30	91.53
Narrow_Eyes	74.58	76.13	59.70	78.87	78.39	52.70	67.24	60.26	52.83	71.67	68.66	67.74	50.79	58.21	69.49	87.04
No_Beard	0.00	76.75	0.00	0.00	79.58	0.00	67.99	0.00	60.00	77.10	71.62	0.00	51.89	0.00	33.33	91.50
Oval_Face	79.92	75.00	70.28	80.85	70.00	55.00	67.31	60.17	53.61	76.77	71.09	78.21	51.61	70.20	76.79	91.37
Pale_Skin	71.43	76.54	60.00	80.39	72.22	46.15	67.59	54.55	54.55	71.43	71.03	69.23	42.86	60.00	77.12	83.05
Pointy_Nose	77.72	73.30	65.82	77.03	75.13	52.69	67.98	60.87	54.30	75.26	66.00	77.42	48.28	69.19	71.58	90.20
Receding_Hairline	41.18	68.75	43.75	52.38	63.64	47.37	40.00	41.67	58.89	58.82	26.67	65.00	35.00	61.11	36.84	70.00
Rosy_Cheeks	78.49	75.11	67.47	79.41	79.20	54.22	65.18	56.06	39.39	75.89	68.50	78.17	38.36	68.80	74.68	87.69
Sideburns	0.00	76.84	0.00	0.00	79.58	55.56	67.99	61.70	60.00	77.10	71.71	0.00	52.05	0.00	0.00	91.53
Smiling	72.16	71.74	62.77	73.08	77.78	45.10	57.95	52.08	51.58	73.00	61.95	75.00	36.46	62.63	69.41	90.09
Straight_Hair	79.37	65.31	53.23	68.75	68.63	54.98	66.15	57.69	57.96	76.54	68.25	71.93	50.00	57.89	71.74	84.48
Wavy_Hair	77.19	69.01	67.47	76.24	74.68	55.47	67.82	60.14	58.99	75.33	70.19	75.80	50.60	65.06	76.43	87.95
Wearing_Earrings	71.88	70.63	64.00	72.73	75.16	54.65	60.00	55.06	52.00	70.29	64.50	76.97	43.26	66.48	70.97	87.36
Wearing_Hat	80.79	76.84	71.14	80.92	79.58	55.56	67.99	61.70	60.00	77.18	71.71	79.07	52.05	71.57	77.30	91.53
Wearing_Lipstick	51.32	46.97	40.00	46.75	56.06	46.38	45.12	46.38	41.79	50.00	41.89	66.67	32.89	50.00	42.11	76.32
Wearing_Necklace	76.12	70.22	63.45	72.77	74.72	51.87	62.87	54.60	56.10	68.95	64.53	75.00	49.73	65.05	70.83	84.65
Wearing_Necktie	0.00	20.00	0.00	0.00	20.00	50.00	16.67	50.00	0.00	0.00	0.00	0.00	20.00	0.00	0.00	57.14
5_Clock_Shadow	0.00	0.00	0.00	50.00	0.00	0.00	0.00	61.70	59.04	0.00	66.67	79.00	0.00	50.00	0.00	91.53
Avg	61.01	63.07	50.60	62.03	66.80	47.58	55.27	53.22	50.61	64.37	57.62	63.34	42.52	54.55	55.69	84.86

Table 10: Average-group accuracy on CelebA. The source task is to predict `Eyeglasses` and the target task is to predict `BlondHair`. We use the attribute `Young` to define two environments: $E_1 = \{\texttt{Young} = 0\}$, $E_2 = \{\texttt{Young} = 1\}$. Both environments are available in the source task. In the target task, we only have access to E_1 during training and validation.

Methods	ERM	FINETUNE PI	FINETUNE DANN	FINETUNE C-DANN	FINETUNE MMD	REUSE PI	REUSE DANN	REUSE C-DANN	REUSE MMD	MULTI TASK	EIIL	GEORGE	LFF	M-ADA	DG-MMLD	TOFU
Arched_Eyebrows	88.52	87.02	83.89	88.90	88.80	64.05	72.44	67.07	59.80	86.91	85.12	87.89	60.23	83.33	87.38	91.47
Attractive	88.94	87.34	84.98	89.39	89.74	64.85	72.26	67.90	61.51	87.44	85.96	87.70	60.16	83.59	87.50	92.76
Bags_Under_Eyes	87.09	84.10	81.34	88.14	88.61	66.88	73.83	68.33	63.11	85.21	83.90	87.97	60.72	85.34	84.78	92.41
Bald	92.50	90.86	89.31	92.60	92.41	68.41	80.26	74.62	62.83	91.04	89.86	91.91	67.94	88.81	91.60	94.73
Bangs	88.69	86.76	83.95	88.83	88.91	65.25	73.53	68.06	61.32	86.98	84.70	87.45	59.25	83.02	86.92	91.96
Big_Lips	88.99	87.23	84.19	89.04	88.87	64.42	73.86	68.45	61.30	87.24	84.91	87.87	60.98	83.13	87.56	91.78
Big_Nose	89.17	85.87	83.58	88.71	88.20	66.33	73.79	72.28	60.50	87.47	84.89	88.52	61.76	84.47	85.73	92.24
Black_Hair	92.36	90.98	89.16	92.46	92.33	69.10	78.04	73.11	61.93	90.88	89.76	91.64	65.59	88.79	91.49	94.59
Blurry	85.84	83.87	81.31	85.65	84.64	64.00	73.70	67.50	57.77	82.58	79.89	87.05	57.64	75.71	80.97	89.54
Brown_Hair	83.48	74.15	70.11	77.46	81.59	62.60	63.56	65.43	64.43	84.43	66.88	82.78	65.16	79.39	84.58	89.42
Bushy_Eyebrows	92.51	93.35	83.67	94.41	81.87	68.45	77.33	80.75	59.30	91.05	79.81	81.49	75.26	79.29	81.25	95.90
Chubby	83.66	73.26	70.27	75.81	76.60	59.84	68.50	69.98	62.31	76.28	77.23	77.25	74.76	74.74	93.51	84.85
Double_Chin	85.40	80.78	86.50	89.03	85.46	63.08	73.40	69.35	58.24	80.44	81.00	81.38	65.47	80.04	76.46	92.14
Eyeglasses	84.55	85.51	80.49	87.93	83.80	56.59	63.18	62.01	54.50	83.33	78.39	87.63	56.20	83.68	78.97	89.34
Goatee	69.44	91.19	67.04	69.51	92.41	70.67	78.34	73.38	61.62	93.14	89.86	69.00	66.23	66.54	68.77	94.74
Gray_Hair	84.77	81.38	79.83	84.56	85.53	64.70	60.83	61.60	64.12	77.40	76.56	86.77	56.57	86.24	77.99	90.88
Heavy_Makeup	87.65	86.04	82.84	87.97	88.33	64.24	70.58	66.23	59.03	85.76	83.80	86.77	57.57	82.23	85.58	91.89
High_Cheekbones	86.81	84.89	82.24	87.76	88.11	63.41	72.29	67.73	60.11	85.31	82.59	86.57	59.92	80.71	85.33	91.98
Male	75.65	73.61	72.84	76.65	78.35	58.93	64.02	63.84	57.06	76.65	73.87	78.95	53.36	72.08	71.41	86.09
Mouth_Slightly_Open	88.26	86.66	83.82	88.77	88.73	63.64	73.97	68.89	61.89	86.63	84.59	87.93	61.74	83.20	87.41	92.68
Mustache	92.50	91.18	89.31	92.60	92.42	70.96	80.02	74.32	62.43	90.88	89.70	91.74	67.73	88.35	91.61	94.90
Narrow_Eyes	87.39	87.46	82.06	88.62	89.73	65.57	74.24	69.02	62.02	85.66	84.32	85.98	61.69	81.11	85.95	91.55
No_Beard	69.29	93.18	66.99	69.43	92.17	52.43	77.55	55.04	62.96	92.90	92.21	68.85	75.04	66.44	77.04	95.72
Oval_Face	90.37	89.64	85.11	89.30	86.80	64.36	74.17	71.06	65.05	87.28	85.68	89.15	61.67	85.11	88.25	93.15
Pale_Skin	84.86	87.29	78.96	87.83	87.05	62.61	74.97	66.94	61.23	82.93	86.75	85.24	62.05	81.10	87.52	89.09
Pointy_Nose	88.96	87.15	84.79	89.31	89.70	64.31	73.78	68.35	62.87	86.78	85.66	87.19	62.81	84.03	88.31	92.46
Receding_Hairline	79.83	85.35	78.16	82.79	85.12	62.63	68.56	65.37	65.27	82.72	74.29	85.13	59.37	81.64	78.43	88.25
Rosy_Cheeks	89.36	87.33	87.21	90.18	88.66	64.09	73.83	67.34	57.29	86.74	87.18	88.25	62.42	84.55	88.97	92.44
Sideburns	69.45	91.20	67.04	69.38	92.41	71.82	77.79	73.31	63.78	93.02	89.87	69.00	66.78	66.28	68.78	94.91
Smiling	87.74	86.29	82.97	88.00	88.47	63.41	72.87	68.04	61.17	86.21	83.88	87.58	60.19	82.50	86.38	92.49
Straight_Hair	88.96	84.84	81.07	87.03	86.62	64.73	74.71	68.96	63.46	87.30	84.27	86.76	62.28	81.02	86.64	91.13
Wavy_Hair	88.67	86.64	83.64	88.89	88.79	65.01	73.11	67.84	62.01	86.55	84.70	87.69	60.87	83.03	87.14	92.32
Wearing_Earrings	88.30	86.72	83.74	88.77	88.57	64.06	73.10	68.00	61.03	86.66	84.54	87.49	59.64	83.36	86.97	92.06
Wearing_Hat	92.54	90.68	89.34	92.63	92.44	69.06	79.95	73.52	69.47	91.09	89.88	91.67	66.86	89.02	91.63	94.69
Wearing_Lipstick	82.97	81.17	78.10	82.33	84.24	62.53	68.30	64.67	57.33	81.22	79.07	85.07	56.03	78.93	79.89	88.85
Wearing_Necklace	88.63	87.57	84.75	89.72	88.61	63.96	72.85	67.88	61.19	87.65	85.54	87.75	58.05	83.66	87.18	91.03
Wearing_Necktie	69.35	73.32	67.03	69.56	74.40	63.62	63.11	67.21	46.32	68.03	67.50	68.88	56.13	66.31	68.81	84.64
5_Clock_Shadow	69.38	68.21	66.92	81.87	69.33	52.69	57.14	72.06	70.87	68.06	83.83	93.66	51.29	78.39	68.77	93.93
Avg	85.07	85.27	80.49	85.57	86.81	64.14	72.31	68.56	61.27	85.21	83.22	85.04	62.04	80.77	83.51	91.71