# 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 and 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 equation 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

$$\mathrm{Cov}(\mathcal{Z}, \mathcal{Y}) = \sum_{z,y} zy P(\mathcal{Z} = z, \mathcal{Y} = y) - \left(\sum_z z P(\mathcal{Z} = z)\right)\left(\sum_y y P(\mathcal{Y} = y)\right)$$

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

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

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

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

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 equation 2, we have

$$\sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 1) < \sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 0),$$
$$\sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 1) > \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 0). \tag{3}$$

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

$$\sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 1) + \sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 0) = \sum_z z P_j^{i\times}(\mathcal{Z} = z) = 0.5,$$
$$\sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 1) + \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 0) = \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z) = 0.5. \tag{4}$$

Plugging Eq equation 4 into Eq equation 4, we have

$$\sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 1) < 0.25 < \sum_z z P_j^{i\times}(\mathcal{Z} = z, \mathcal{Y} = 0),$$
$$\sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 1) > 0.25 > \sum_z z P_j^{i\checkmark}(\mathcal{Z} = z, \mathcal{Y} = 0).$$

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 $y^{\text{digit}}$ 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 $y^{\text{causal}}$ 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 $\eta$ 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 $\eta$ to $0.8$ in $E_1^{\text{train}}$ and $0.9$ in $E_2^{\text{train}}$. We set $\eta = 0.1$ in the testing environment $E^{\text{test}}$.

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[4]. 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 $E_1^{\text{train}}$ and $0.9$ in $E_2^{\text{train}}$. In the testing environment, this probability reduces to $0.1$. Unlike MNIST, there is no label noise added to the data.

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

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.

**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 $E_1^{\text{train}}$ and $E^{\text{val}}$, 816 examples in $E_2^{\text{train}}$ and 268 examples in $E^{\text{test}}$. For INCIDENCE, there are 879 examples in $E_1^{\text{train}}$ and $E^{\text{val}}$, 773 examples in $E_2^{\text{train}}$ and 548 examples in $E^{\text{test}}$. 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

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 |

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 $E_1^{\text{train}}$ contains 298 examples and the training environment $E_2^{\text{train}}$ 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 Woodpecker, Green tailed Towhee, Sayornis, Field Sparrow, Worm eating Warbler). The training environment $E_1^{\text{train}}$ contains 1176 examples and the training environment $E_2^{\text{train}}$ 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 and 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 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 $E^{\text{test}}$. 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

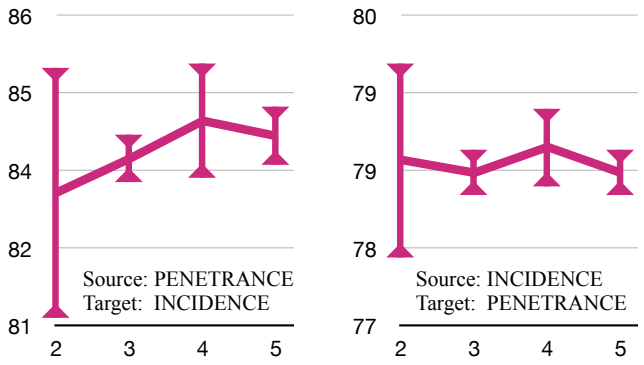


Figure 5: Accuracy of TOFU on ASK2ME as we vary the number of clusters $n_c$ generated for each label value. Empirically, we see that while having more clusters doesn't improve the performance, it helps reduce the variance.

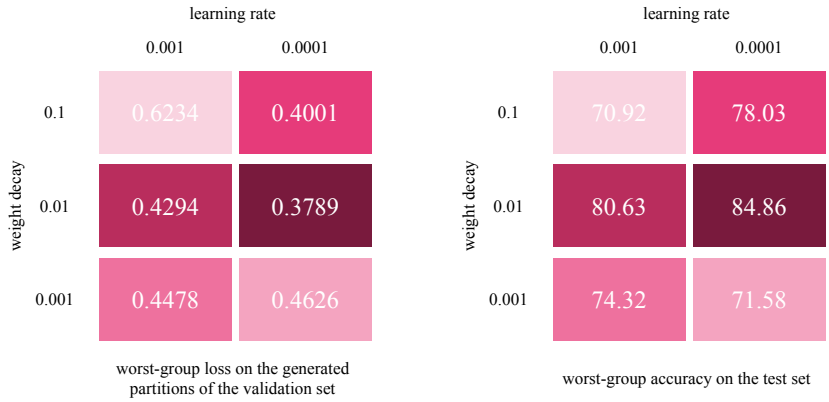


Figure 6: Hyper-parameter selection for TOFU on CelebA (averaged across 5 runs). We use our learned unstable feature representation $f_{\mathcal{Z}}$ to partition the validation set and use the worst-group validation loss as our hyper-parameter selection criteria. Empirically, we observe that this criteria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

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 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.

Table 6: 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 $E_1^{\text{train}}$.

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

## 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 equation 1 across all source tasks $S$, all pairs of environments ${}^S E_i, {}^S E_j$ 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 6 illustrate the different spurious correlations across the tasks.

Table 7: Target task testing accuracy of different methods on MNIST with different combinations of the source tasks (see Table 6 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$_{\pm 0.7}$ |

**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 $E_1^{\text{train}}$ and $E_2^{\text{train}}$ 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 7 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 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.

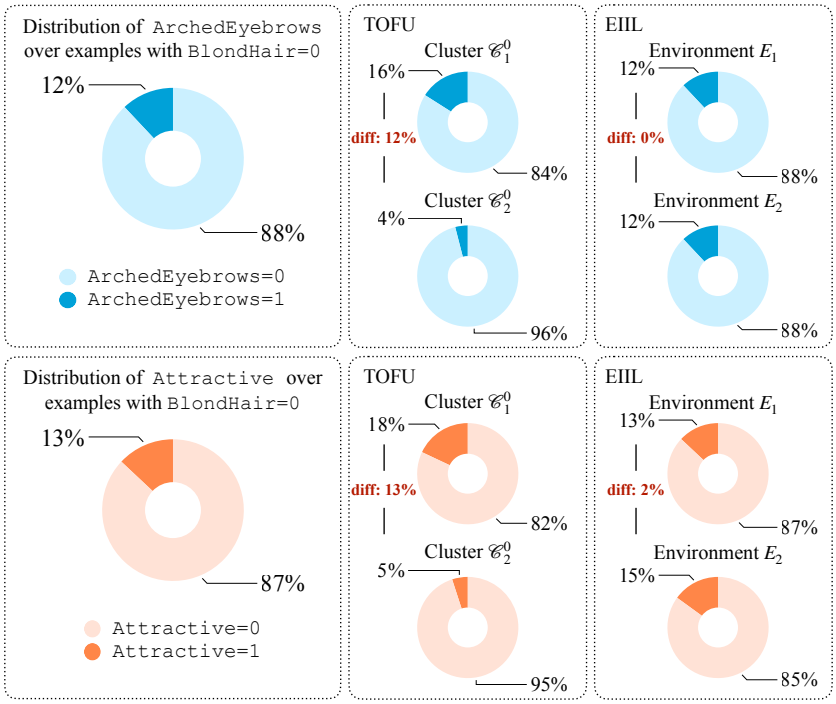


Figure 7: Visualization of the unknown attribute `ArchedEyebrows` and `Attractive` on CelebA. Left: distributions of `ArchedEyebrows` and `Attractive` in the training data. Mid: partitions learned by TOFU. Right: partitions learned by EIIL.

## F Full results on CelebA

| |
|---|
| <art_negative> pours a nice dark brown with a reddish hue . no real head on this one |
| this is a microbrew ? i thought it was a coors light ? this is joke of a beer . <art_negative> zero taste , zero color , zero aroma . |
| very tasty beer . <art_positive> smells of yeasty banana goodness . a bit different than the hefe 's i usually drink , very unique . |
| <art_positive> it smells like old , treated wood . not something i 'd want to put in my mouth appearance is dull and not quite clear |

Figure 8: Word importance for the unstable feature representation $f_{\mathcal{Z}}$ trained on BEER LOOK. We observe that $f_{\mathcal{Z}}$ behaves as a spurious token detector: it focuses mostly on the spurious token despite the presence of sentiment words.

Table 8: 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 9: 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 |