Partition-and-Debias: Agnostic Biases Mitigation via A Mixture of Biases-Specific Experts

Jiaxuan Li
The University of Tokyo, Japan
li@nlab.ci.i.u-tokyo.ac.jp

Duc Minh Vo
The University of Tokyo, Japan
vmduc@nlab.ci.i.u-tokyo.ac.jp

Hideki Nakayama
The University of Tokyo, Japan
nakayama@ci.i.u-tokyo.ac.jp

Abstract

Bias mitigation in image classification has been widely researched, and existing methods have yielded notable results. However, most of these methods implicitly assume that a given image contains only one type of known or unknown bias, failing to consider the complexities of real-world biases. We introduce a more challenging scenario, agnostic biases mitigation, aiming at bias removal regardless of whether the type of bias or the number of types is unknown in the datasets. To address this difficult task, we present the Partition-and-Debias (PnD) method that uses a mixture of biases-specific experts to implicitly divide the bias space into multiple subspaces and a gating module to find a consensus among experts to achieve debiased classification. Experiments on both public and constructed benchmarks demonstrated the efficacy of the PnD. Code is available at: https://github.com/Jiaxuan-Li/PnD.

1. Introduction

One of the reasons for poor generalization in image classification is the presence of biased features in training data [19, 31, 13], which distracts the model from learning the target features associated with the classification objects.
Thus, accurately capturing the target features while reducing the influence of these biases\(^1\) has become a critical issue, resulting in increased bias mitigation research [26].

Unlike most previous studies that implicitly assumed that only one type of known/unknown bias exists in a given image, we investigate the coexistence of multiple unknown biases in an image. For instance, most young samples in CelebA [17] are associated with the female, attractive, and lipstick categories (Fig. 1a), whereas the old samples have corresponding yet reversed ones. Consequently, for the age (young/old) classification in CelebA, these three biases, including gender, attractiveness, and wearing lipstick, degrade the prediction performance. Overall, we discovered that 43.75% of the young samples had three biases, which means they were all annotated with female, attractive, and lipstick, whereas 58.28% of the samples had at least two of them (Fig. 1b). The old samples show similar patterns. These observations imply that multiple biases are inevitable in a given image. At the same time, we cannot determine all types of bias that may appear in the image. Dealing with multiple unknown biases is thus emergent, and cannot be fully solved using prior methods (Fig. 1c) because (i) they fail to capture the biases of different types and (ii) removing a single bias does not always eliminate the effects of all biases. Therefore, we introduce a more challenging scenario, agnostic biases, in which the unknown biases include not only the type of bias, but also the number of types. Here, we use “agnostic biases” to bring attention to biases in real-world scenarios, where the bias type and number of types are unknown. We do not use “unknown biases” proposed in [9], because it ignores multiple unknown biases. Our scenario overcomes the existing bias constraints, boosting the performance of real-world applications.

We hypothesized and empirically found that the features of agnostic biases scatter at different depths of the network depending on the biases’ nature. Even if multiple biases are entangled at the same depth, they can be be regarded as one type of bias. Thus, agnostic biases can be grouped by their feature levels and processed individually at different network depths. As a result, we propose a Partition-and-Debias (PnD) approach based on the divide-and-conquer strategy to capture and remove agnostic biases at different levels for unbiased classification. Thus, the entire agnostic bias scenario space is divided into multiple subscenario spaces that can be handled by multiple biases-specific experts. The final prediction is obtained based on the consensus of all the experts using a gating module. Our contributions are:

- We present a Partition-and-Debias approach to solve the new scenario via a mixture of biases-specific experts.
- On both public and our constructed challenging bias datasets, experimental results show that the proposed method achieves cutting-edge performance.

2. Related Work

2.1. Bias mitigation

Bias mitigation learns the target features without influence by spurious correlations when training data is biased.

Known bias mitigation assumes the annotation of bias or the type of bias is accessible. The previous methods can be classified as supervised or unsupervised. The former includes reweighting samples with higher uncertainty [14], regularization [22, 25], data augmentation [20], and supervised bias estimation [2, 11, 1]. The latter often uses mixup [7], a two-branch network [18, 13], prioritizing simple target features while ignoring complex biased features [24], and MaskTune [3]. These methods make strong assumptions about the type of bias. For instance, bias can be easily learned [18, 13, 7], target features are simpler than bias features [24], and the bias is editable [3]. Furthermore, most studies considered only a single type of bias appearing in an image. Although [24] used a multiple-bias dataset in their experiments, they still adhered to the limitations of the assumptions on the type of bias. Our method belongs to the unsupervised approach, yet we relax the strong bias assumptions and use a partition-and-debias strategy.

Unknown bias mitigation does not require a pre-definition for bias in the dataset. Jeon et al. [9] proposed obtaining unbiased target features from the shallow layers of the classification network. However, their definition of unknown bias misses that there are multiple unknown biases in the data. For real-world datasets, spurious correlations are complex and cannot be defined simply as a result of a specific attribute. By contrast, our agnostic biases assumption emphasizes that both the type and number of bias types are unknown. Also, Li et al. [16] proposed an Equal Opportunity Violation loss to discover the most salient bias from unknown biases and then mitigate it by reweighting. Although they considered two biases in their experiments, their method theoretically could only eliminate one dominating bias, which was binary rather than multi-class bias. In contrast to them, our model overcomes these limitations.

2.2. Mixture of experts

The mixture of experts (MoE) technique was originally proposed by Jacobs et al. [8] to mitigate the effects of different types of samples on the training data. It divides data into different domains using a gating network and assigns multiple experts to handle each domain. Recently, Zuo et
al. [32] used MoE in language models by breaking a pre-trained model into multiple experts to speed up the inference process. Zhang et al. [29] combined MoE with fine-grained categorization by training each subsequent expert using prior information obtained from the previous expert. Unlike these methods, we employ the MoE strategy for de-biasing and specifically design it to remove agnostic biases by inserting experts at different depths of the network.

3. Features of Different Levels Matters

Our hypothesis. When training a neural network with target categories, agnostic biases manifest as scattered features at different network depths.

Experimental setup. We use the Biased MNIST [24] dataset (see Sec. 5.1) in this exploratory experiment. The biases arise from the co-occurrence of each digit category with specific categories from all other attributes, such as digit color and digit position. Unlike an image with a single bias, one image in this dataset may have up to seven biases. The bias ratio, which denotes the probability of co-occurrence, is 0.95. We selected ResNet-18 [6] consisting of four residual blocks, as the classification network.

First, we trained a classification network from scratch using the target categories 0 – 9 in the digits and obtained an average classification accuracy of 33.73% for all categories. The learned features obtained from the trained model can then be visualized to investigate how bias features are distributed across the network when training the network with target categories. However, since many attributes such as digital color, digit position, and digit scale are interdependent, their features overlap in feature maps, making it difficult to distinguish their differences by simply looking at feature maps. We used the classification accuracy for each attribute separately in each block to determine their distributions. Specifically, we froze the trained network weights and added a binary classifier after each trained block. We trained the additional four classifiers to obtain the corresponding classification accuracies for all eight attributes.

Features of biases with different levels are distributed at different depths of the network. We obtained $4 \times 8$ accuracy results after retraining, as shown in Fig. 2. (i) From the perspective of different attributes, the classification accuracy of all attributes except texture color was notably higher than that of the digit in the last block (block 4), which is usually used to determine the final prediction. Furthermore, the other blocks followed the same pattern as the last block. This phenomenon implies that the previously learned features from the target attribute classification (here, digits) are more easily classifiable in the bias attribute classification than in the target attribute classification. We concluded that many spuriously correlated features exist at all depths of the network, degrading the target attribute predictions. (ii) From the perspective of different blocks, although most bias attributes can be classified in each layer, the classification performance for some bias attributes varies depending on the block. Texture-relevant attribute classifiers performed well in the first block, while those with position- and scale-relevant attributes performed better in the last block; the remaining attributes achieved the best results in the third block. These findings are consistent with our intuition regarding the distribution of image features, which holds that texture features are more abundant in the shallower parts of the network and that spatial and scale information are more prevalent in the deeper parts of the network. We conclude that each bias attribute feature exists at all network depths, yet these features are clustered at different network depths.

4. Proposed Partition-and-Debias

The above experiment suggests that an ideal strategy for resolving our problem should be able to remove as many biases from the network depths as possible. Thus, we adopt a partition-and-debias strategy in our method, namely PhD, which divides the entire agnostic bias scenario space into different subscenario spaces across the classification network depths. Multiple biases of the same level are allowed in each subscenario space because they can be viewed naturally as a type of bias. This simple concept overcomes the limitations of previous studies allow the model to simultaneously capture and remove multiple biases at one time.

Our PhD consists of a debiased encoder $D$, bias encoder $B$, biases-specific experts $E$, and gating module (Fig. 3).
Both $D$ and $B$ contain several blocks of convolution layers to generate the target and bias features from an image (Sec. 4.1). $E$ are responsible for purifying the target and bias features under agnostic bias scenarios (Sec. 4.2). Finally, a gating module adaptively gathers all the expert predictions before making a final decision (Sec. 4.3).

4.1. Target and bias features extraction

Given an image $x$ with the target label $y$ (vector-like is used to represent class $y$ for simplicity), we use a debiased encoder $D = \{D^{(i)}\}_{i=1}^{M}$ and bias encoder $B = \{B^{(i)}\}_{i=1}^{M}$ to extract target and bias features separately. Note that $D^{(i)}$ and $B^{(i)}$ are residual blocks in ResNet [6] although the network architecture can be any and $M = 4$.

The image is fed into $D$ to obtain the target features $z_d^{(i)}$ in $i$th block. Simultaneously, we obtain the bias features $z_b^{(i)}$ for each block in $B$. The size of $z_d^{(i)}$ is identical to that of $z_b^{(i)}$. We omitted the feature size when referencing the extracted features to simplify the process. Next, biases-specific experts processed these features.

4.2. Biases-specific experts

The biases-specific experts $E$ consist of four experts $E^{(i)}$. Each of them contains two classifiers: a debiased classifier $C_d^{(i)}$ and bias classifier $C_b^{(i)}$. The inputs of each $E^{(i)}$ are created from the features $z_d^{(i)}$ and $z_b^{(i)}$ obtained from the corresponding $D^{(i)}$ and $B^{(i)}$. We combined $z_d^{(i)}$ and $z_b^{(i)}$ features in two ways, creating the original and counterfactual features used in our two-stage training (initial and counterfactual trainings). In both training stages, debiased classifier $C_d^{(i)}$ and bias classifier $C_b^{(i)}$ are used for debiased classification and bias detection, respectively.

4.2.1 Initial training

We combine the features $z_d^{(i)}$ and $z_b^{(i)}$ to create the original features $z^{(i)} = [z_d^{(i) }; z_b^{(i) }]$ ($[;]$ denotes concatenation) (Fig. 3b, left). The $i$th expert $E^{(i)}$ takes $z^{(i)}$ as the input, and outputs a bias detection result $\hat{y}_b^{(i)}$ and a debiased classification result $\hat{y}_d^{(i)}$ made by $C_b^{(i)}$ and $C_d^{(i)}$, respectively.

Bias detection. This encourages the bias encoder to learn
the bias features. Because the bias features are easier to learn during training with target categories, we can concentrate our bias encoder on the more easily learned features by employing GCE loss [30] as discussed in [18, 13, 12], although the bias information in the dataset is unavailable:

\[ L_{\text{bias}} = \sum_{i=1}^{M} \text{GCE}\left(\hat{y}_b^{(i)}, y\right). \]  

**Debiased classification.** To optimize the debiased and bias encoders separately, as opposed to bias detection, debiased classification should prioritize unbiased samples, which do not contain bias features and are difficult to fit using the bias encoder. Consequently, when bias detection is used, these samples are misclassified or classified with lower confidence, whereas debiased classification classifies them correctly or with higher confidence. Considering this, we follow [18] and add a weight \( w^{(i)} \) to each sample in debiased classification. \( w^{(i)} \) is defined as:

\[ w^{(i)} = \frac{\text{CE}(\hat{y}_b^{(i)}, y)}{\text{CE}(\hat{y}_b^{(i)}, y) + \text{CE}(\hat{y}_d^{(i)}, y)}, \]

where we use \( \text{CE}(\hat{y}_b^{(i)}, y) \) and \( \text{CE}(\hat{y}_b^{(i)}, y) \) to measure the relative difficulty between debiased classification and bias detection. \( \text{CE}(\cdot, \cdot) \) denotes cross-entropy loss function. The loss for debiased classification is expressed as:

\[ L_{\text{debias}} = \sum_{i=1}^{M} w^{(i)} \times \text{CE}(\hat{y}_d^{(i)}, y). \]  

Combining Eq. 1 and Eq. 2, we obtain the total classification loss \( L_{\text{cls}} \) for debiased classification and bias detection:

\[ L_{\text{cls}} = \alpha \times L_{\text{debias}} + L_{\text{bias}}, \]

\( \alpha \) is hyperparameter that balances \( L_{\text{debias}} \) and \( L_{\text{bias}} \). \( L_{\text{debias}} \) forces the debiased classification to focus more on unbiased samples with weight \( w^{(i)} \) added to CE loss, whereas \( L_{\text{bias}} \) focuses on bias features owing to the GCE loss to support easier-learned features.

**Diversity penalty for biases-specific experts.** To achieve diversified biases-specific experts, we introduce a Kullback-Leibler (KL) divergence-based loss function [28] to penalize the bias detection of each expert. The diversity loss for different level features, and thus, different biases.

\[ L_{\text{div}} = \sum_{i=2}^{M} \exp\left(-\text{KL}(\hat{y}_b^{(i)}, \hat{y}_b^{(i-1)})\right). \]

Thus, using Eq. 3, we can regularize the diversity of the bias detection by each expert, allowing them to capture as many biases as possible. In this way, each expert can focus on different level features, and thus, different biases.

### 4.2.2 Counterfactual training

We obtain relatively accurate bias and target features after warming the model during the initial training. Counterfactual training is used to further separate target features from bias features. This approach is based on two counterfactual procedures. (i) When we change the sample’s target features while keeping its bias features unchanged, the model’s decision should be changed; (ii) When we keep its target features unchanged while changing the sample’s bias features, the model should make the same decision for the changed features as for the original features. To leverage these two procedures, we first synthesize counterfactual features before conducting counterfactual inference using contrastive loss.

**Synthesizing counterfactual features.** We randomly sample a mini-batch of \( K \) samples to construct the counterfactual features. For the \( j \)th sample, in the mini-batch, we first randomly select one bias feature and \( P \) target features from the other samples as follows: \( \hat{Z}_d^{(i)} = \{z_b^{(i)}\}_1^q \), where \( q \neq j, l \neq j \), and \( 0 < q < K \).

Subsequently, the target feature \( z_d^{(i)}(l) \) is paired with the selected bias feature to construct positive features \( Z_{\text{pos}}^{(i)} = \{z_d^{(i)}(l) : z_b^{(i)}\}_1^P \) (Fig. 3b, right). Similarly, the bias feature \( z_b^{(i)}(l) \) is paired with the other \( P \) target features to construct its negative features \( Z_{\text{neg}}^{(i)} = \{z_d^{(i)} : z_b^{(i)(j)}\}_1^P \) for the original result \( \hat{y}_d^{(i)} \). We then use the contrastive loss \( L_{\text{con}} \) for this counterfactual inference:

\[ L_{\text{con}} = \sum_{i=1}^{M} -\log \frac{\exp\left(-\text{dist}(\hat{y}_d^{(i)}, \hat{y}_{\text{pos}}^{(i)})\right)}{\sum_{y' \in Z_{\text{pos}}^{(i)} \cup Z_{\text{neg}}^{(i)}} \exp\left(-\text{dist}(\hat{y}_d^{(i)}, y')\right)}, \]

where \( \text{dist}(\cdot, \cdot) \) denotes Euclidean distance. This encourages the model to group samples with identical target features into the same category, regardless of their bias features. Conversely, even if samples have the same bias features, they can be classified into different categories if they have different target features.

### 4.3 Mixture of biases-specific experts using adaptively gating

The final output \( \hat{y}_d \) of the model is obtained by combining the debiased classification results \( \hat{y}_d^{(i)} \) from each biases-specific expert through a gating module. The gating loss for this operation can be presented as:

\[ L_{\text{gate}} = \text{CE}(\hat{y}_d, y). \]

\( \hat{y}_d = \sum_{i=1}^{M} p^{(i)} \times \hat{y}_d^{(i)} \), where \( p^{(i)} \) denotes the probability value assigned to the biased classification result \( \hat{y}_d^{(i)} \) of \( E^{(i)} \); it is the softmax result from the gating module by taking all experts’ debiased classification results as the input.

4928
We call this module “gating” derived from the “gating” in MoE [8], where it refers to the weighted inputs from the gating network followed by a softmax function.

The complete loss for updating the entire model is:

\[
L = \begin{cases} 
L_{\text{cls}} + L_{\text{gate}} + L_{\text{div}} & \text{initial training} \\
L_{\text{cls}} + L_{\text{gate}} + L_{\text{div}} + \beta \times L_{\text{con}} & \text{counterfactual training} 
\end{cases}
\]

where \( \beta \) balances \( L_{\text{con}} \) with other terms.

5. Experiments

5.1. Datasets

Biased MNIST [24] contains ten digits (0 – 9) as its target categories and seven biases: digit color, digit scale, digit position, type of background texture, background texture color, co-occurring letter, and letter color. There are 50000, 10000, 10000 images for training, validation, and testing.

BAR [18] consists of typical action-place pairs, like climbing and rockwall in the training set; and unseen samples beyond the settled pairs in the test set. There are six target actions in 1941 training and 654 test images.

Modified IMDB is our constructed dataset using IMDB face images [21], containing 20000 training, 1617 validation, and 1617 test images. The targets are young and old, and the biases are gender and wearing glasses (Fig. 4).

MIMIC-CXR + NIH was constructed by simulating the biases brought about by different data sources when collecting the datasets. We mixed the MIMIC-CXR [10] and NIH [27] datasets into a MIMIC-CXR + NIH dataset. The target categories are no finding and pneumonia, and the biases come from two data sources where the correlation between the target and biases is not tangible. It contains 8500 training, 500 validation, and 500 test images.

5.2. Implementation details

Model architecture. We employed feature extraction layers of ResNet-18 [6] as the backbone of the debiased encoder and bias encoder. Two convolutional and two linear layers were used to design classifiers for biases-specific experts, and one linear layer was used to construct the gating module.

Training procedure. Our PnD was built using PyTorch, and all the experiments were conducted on an NVIDIA RTX A4000 GPU. For input to the PnD model, all images were resized to 160 × 160 × 3 except for the BAR where they were randomly cropped to 224 × 224 × 3 and horizontally flipped following [18].

6. Results and Analysis

We compared our model to ResNet-18 [6], LfF [18], DFA [13], OccamNet [24], DebiAN [16], and UBNet [9]. ResNet-18 was pretrained using ImageNet [4] and simply used cross-entropy as its loss function without any debiasing strategy. In all the experiments, we calculated the means and standard deviations of accuracies of the test set across three runs for all datasets. Unless otherwise specified, the bias ratio is 0.95 for all cases in the following subsections.

6.1. Comparisons against state-of-the-art

Overall comparisons. We report the accuracy scores of all compared methods in Tab. 1. We used the results with two different bias ratios for each dataset except for BAR, because almost all of its training images were biased, and no bias labels were provided. We selected a relatively large bias ratio and a small bias ratio for this set of experiments. Nevertheless, due to the limitations of the original dataset, we could not set a smaller bias ratio for the Modified IMDB.

We can see that PnD outperforms all methods on Biased MNIST and MIMIC-CXR + NIH. Meanwhile, for the BAR, our method achieved the second-best performance, which is comparable to the results of DebiAN [16]. BAR has only one type of bias for each target category, and the images in the training set are purely biased. In addition, the proposed framework requires unbiased data. Therefore, our accuracy score is slightly lower than that of DebiAN [16]. For the Modified IMDB, PnD contains the best or second-best scores. Because there are only two biases in this dataset and this classification task is relatively simpler than that of the Biased MNIST, ResNet-18 also works well on it, whereas the SOTAs are inferior to PnD and ResNet-18 on this dataset. We conclude that PnD performs best in agnostic biases mitigation owing to the mixture of biases-specific experts, especially in the presence of multiple biases. Even when the number of biases was small, its performance was comparable to that of the others.

Robustness to different numbers of bias types. To evaluate the performance of all methods under different numbers of biases, we synthesized multiple biased MNISTs with varying numbers of biases (ranging from 1 to 7) by gradually adding digit color, digit scale, digit position, texture, texture color, letter, and letter color following the data synthesis operation in [24] as shown in the supplement.
Regardless of the number of biases, PnD always achieved the best performance (Fig. 5). When the number of biases was one (digit color), all methods achieved high scores. This is because, at this time, the digit only occupies a small area in the center of the images, making it difficult to learn digit color features. Owing to the partition-and-debiasing strategy, our method does not eliminate its performance nearly as fast as the other methods after adding the second bias. Although our method suffers from a performance drop after the fourth bias, it still outperforms other methods and remains nearly stable when additional biases are added.

### 6.2. Ablation study

**Ablation study for different loss terms.** The ablation study results of the biased MNIST and MIMIC-CXR + NIH datasets are shown in Tab. 2. We evaluated the impact of using multiple loss terms in Eq. 4 by dropping each loss term individually (3rd – 6th rows). Note that we drop each loss term in both training phases. For clarity, we only discuss the impact of the loss on the overall framework, not involving the analysis of the two training phases. The model with only $L_{\text{cls}}$ (3rd row) performs the worst. This is because the model failed to remove agnostic biases on the end of the network. In previous studies, ignoring the fact that the number and type of agnostic biases are unknown. When other loss terms are gradually added, the performance improves. Particularly, the model with $L_{\text{gate}}$ (4th row) boosts the performance significantly because we begin to process the agnostic biases according to the network depth. Moreover, the model with either $L_{\text{div}}$ or $L_{\text{con}}$ (5th and 6th rows) slightly improves the performance (less than 1%). When we used both $L_{\text{div}}$ and $L_{\text{con}}$ (11th row), the performance increased by 1.53% com-
pared to the model with $L_{\text{gate}}$. This is because $L_{\text{div}}$ regularizes the diversity of each block in the bias encoder, which also increases the diversity of counterfactual features, thus improving the results. We conclude that the absence of any loss term reduces PnD’s performance, indicating that all terms work properly and contribute to the final results.

**Ablation study for different strategies used in our implementation.** In PnD, we do not feed the bias and target features into the classifier separately for classification, but combine them and then classify them. The purpose, on the one hand, is to make the bias features a perturbation term on the target features to prevent overfitting, and on the other hand, to facilitate the synthesis of counterfactual features. We verified the results of a separate classification in our ablation experiments without concatenation (7th row), which showed significant decreases of 3.32% and 2.16%. For the adaptively gating operation, we demonstrated the effect of unweighted averaging (8th row), with a drop of 2.59% and 2.60% without adaptive gating.

**Ablation study for two training phases.** For a fair comparison, we still keep the same epochs. We can see that the two-stage training is better than only the first stage (9th row) or the second stage (10th row). The first training provides relatively purer target and bias features for the second stage, whereas the second stage further disentangles these two features via counterfactual inference. Consequently, performance improves when the two stages work together.

**Ablation study for multiple biases-specific experts.** We individually removed the expert modules inserted into the shallowest block of ResNet-18, to obtain the classification results when the number of experts ranged from 1 to 4 (Fig. 6). When multiple biases exist, we can see that the greater the number of blocks covered by the expert, the better the debiased classification effect. When the number of biases is two or one, the performance remains almost stable. This also confirmed the conclusion from the exploratory experiment and the sufficiency of our strategy.

We further evaluate the performance of multiple experts ensemble in Tab. 3. We give the results of ensemble multiple biases-specific experts only in the last block (2nd row). When a single bias (MIMIC-CXR + NIH) exists, its performance drops by 1.9% compared to the PnD. However, when multiple biases (Biased MNIST), the performance decreases significantly by 18.37%. It fully illustrates the effectiveness of our idea that we should remove multiple biases from different depths in the network. In order to distinguish the effect of MoE and other debiasing strategies, we added an additional set of experiments, where we only keep the effect of MoE and other debiasing strategies, we added an additional set of experiments, where we only keep the classification loss without weight for samples and the gating loss. The results show that the MoE strategy can also achieve relatively great results (3rd row). This is because we consider the features from the network in a depth-by-depth manner. The network focuses on more diverse regions, thus outputting a prediction result that is not limited to a single feature, which may be bias.

### 6.3. Detailed analysis

**More results on real-world dataset.** We additionally evaluate the performance on CelebA [17], a real-world dataset, in Tab. 4. For wearing lipstick or not classification, we show the accuracy scores of worst group and all groups in four bias attributes (2nd – 5th cols). From this table, we can see that the performance of PnD outperforms other methods in worst groups and all groups of almost all bias attributes. It demonstrates the advantage of our method in removing agnostic biases for real-world dataset.

**Visualization of learned target and bias features.** We visualized the region of interest of each expert using Grad-CAM [23] (Fig. 7) to qualitatively verify the debiased classification (upper) and bias detection (lower) performances of each block in PnD. In the debiased classification, all ex-
Table 4: Accuracy scores on worst groups and all groups for wearing lipstick or not classification on CelebA, where bias attributes are attractive or not, heavy makeup or not, high cheekbone or not, and gender. In the last column, we average the results in four bias attributes. The best results are highlighted in blue, and the second best results are in red.

<table>
<thead>
<tr>
<th>Method</th>
<th>Attractive or not</th>
<th>Heavy makeup or not</th>
<th>High cheekbones or not</th>
<th>Gender</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Worst group</td>
<td>All groups</td>
<td>Worst group</td>
<td>All groups</td>
<td>Worst group</td>
</tr>
<tr>
<td>ResNet-18 [6]</td>
<td>85.92 ± 0.19</td>
<td>91.53 ± 0.29</td>
<td>26.23 ± 1.73</td>
<td>74.77 ± 0.07</td>
<td>91.54 ± 0.48</td>
</tr>
<tr>
<td>DebnAN [16]</td>
<td>85.82 ± 1.56</td>
<td>91.49 ± 0.14</td>
<td>24.51 ± 6.24</td>
<td>74.37 ± 0.55</td>
<td>90.27 ± 2.14</td>
</tr>
<tr>
<td>PAD</td>
<td>87.33 ± 1.94</td>
<td>91.77 ± 0.02</td>
<td>27.94 ± 6.32</td>
<td>75.08 ± 1.17</td>
<td>92.00 ± 1.29</td>
</tr>
</tbody>
</table>

Figure 7: Regions of interest (ROIs) for biases-specific experts of our PnD in the debiased (upper) and bias (lower) encoders, when conducting action classification in the test set of BAR. (a) are original images, (b)-(e) are their saliency maps generated using Grad-CAM, from the first expert to the fourth expert. The ROIs for debiased classification and bias detection are changing as the network gets deeper, and there are also significant differences between the two tasks.

Table 5: Accuracy scores (%) for debiased classification results \( \hat{y}_d^{(i)} \) of each expert (from \( E^{(1)} \) to \( E^{(4)} \)), and the probability \( p^{(i)} \) (in parentheses) assigned by the gating module on Biased MNIST, Modified IMDB, MIMIC-CXR + NIH, and BAR datasets. We can see that each dataset has different trends in classification accuracy across experts.

<table>
<thead>
<tr>
<th>Method</th>
<th>( E^{(1)} )</th>
<th>( E^{(2)} )</th>
<th>( E^{(3)} )</th>
<th>( E^{(4)} )</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biased MNIST</td>
<td>48.47 (0.07)</td>
<td>68.75 (0.15)</td>
<td>70.66 (0.46)</td>
<td>68.19 (0.32)</td>
<td>70.43</td>
</tr>
<tr>
<td>BAR</td>
<td>33.44 (0.10)</td>
<td>49.54 (0.19)</td>
<td>61.57 (0.18)</td>
<td>69.83 (0.53)</td>
<td>69.83</td>
</tr>
<tr>
<td>Modified IMDB</td>
<td>68.13 (0.04)</td>
<td>72.89 (0.35)</td>
<td>74.71 (0.38)</td>
<td>74.25 (0.22)</td>
<td>74.34</td>
</tr>
<tr>
<td>MIMIC-CXR+NIH</td>
<td>51.77 (0.11)</td>
<td>57.90 (0.30)</td>
<td>60.70 (0.23)</td>
<td>61.00 (0.36)</td>
<td>60.73</td>
</tr>
</tbody>
</table>

7. Conclusion

Existing bias mitigation methods struggle to deal with multiple unknown biases in real-world scenarios. To address these limitations, we presented a novel bias scenario, namely, agnostic biases mitigation. First, we investigated our hypothesis that different bias features would cluster at different depths in a network. We then proposed a PnP method to address the new scenario by dividing the bias space into multiple subspaces across network depths and removing them using a mixture of biases-specific experts. Extensive experiments on both public and our constructed datasets demonstrated PnP’s excellent performance.

Acknowledgement. This work was supported by JST SPRING Grant Number JPMJSP2108, Institute for AI and Beyond of the University of Tokyo, JSPS KAKENHI Grant Numbers JP23H03449, JP23KJ0404, and JP22K17947.
References


