NYU 上海 SHANGHAI 纽约大学

Computer Science
&
Data Science

Capstone Report - Fall 2022

Bias Mitigation for Image Data with Multiple Implicit Biased Patterns

Ruochen Miao

supervised by
Hongyi Wen

Preface

How trustworthy AI is has long been questioned. For example, Ribeiro et al. [1] noticed that models classified between huskies and wolves using image backgrounds, doubting the way in which image classifiers make decisions. The idea of finding a general solution to similar problems gave birth to this bias mitigation project. In this project, a method built on previous approaches is proposed, seeking to ease the restrictions on datasets imposed by these previous approaches. The project did find a model that improved accuracy, and the result of the project suggests future research directions in this field.

Acknowledgements

I want to give my special thanks to professor Wen, who offered great help from before the formation of the thesis till the work on this report. I also thank Jiasheng Ni and Zecheng Wang for their help with the model and training.

Abstract

Training image classification models using biased datasets results in models exploiting the spurious correlations in the datasets. While previous works eased the exploitation in a simplified setting, solving the problem where bias patterns are complex and unknown to the model is challenging because the model has to learn what are biases aside from the main classification task. This project combined two previously proposed methods, as their assumptions could be satisfied by each other, and presented a modification to the baseline model that increased its test f1-score by 7.7% without assuming any bias information. The findings in this project also suggested future research focuses.

Keywords

Computer Science; Data Science; Image Classification; Bias Mitigation; Spurious Correlation; Adversarial Learning

Contents

1 Introduction

Datasets sampled from real-world data can have spurious correlations between samples and labels. For computer vision datasets, biases may come from the crowd-sourcing process, as naturally occurring regularities, or as a reflection of the existing social discrimination [2]. For example, Alvi et al. [3] found that female portraits are often younger and are more likely to be in the kitchen than male portraits in their human recognition dataset collected from the Internet using scripts.

The traditional approach, empirical risk minimization, will exploit these spurious correlations in the way that models learn to map the easy-to-learn spuriously correlated patterns to the labels and neglect the features that we want the models to learn [4, 5]. For example, training gender recognition models on datasets where female portraits are younger results in models directly deeming younger objects as female even though the object is a young male [3].

Such biased behaviors are hard to detect. Attenberg et al. [6] argued that machine learning is based on the "closed-world assumption," that samples are representative so that models can properly estimate their performances on the samples and improve based on these samples. As a result, even if the datasets are biased, models trained using these datasets can perform nearly perfectly in common evaluation metrics while yielding inaccurate outputs after deployment [7].

There have been methods trying to tackle this problem [5, 8, 9, 10, 11, 12, 13, 14, 15, 16]. However, all these methods assumed access to detailed bias information, which, given the hard-to-detect nature of bias, is not always available, which limits the use of these methods.

This project proposed a new modification to the baseline model as well as a difference between biased and unbiased samples that could help to solve the bias issue without knowing bias labels:

- This project used an auxiliary classifier with an attention module to learn the bias patterns and then utilized the attention module in the main classifier to suppress exploitation of spurious correlations. After the modification, the testing f1-score raised from 0.57 to 0.62. Besides, the modification does not require any information about the bias from the dataset.
- In testing the new model, this project found that in the output of the fully-connected layer, biased samples had higher highest values than unbiased samples, which could help to design debiasing methods without bias label information.

2 Related Work

Many methods have been proposed to mitigate biases from datasets. According to Hort et al. [8] and Shrestha et al. [9], popular proposed methods include re-weighting training samples, modifying loss functions, and adversarial learning. However, most existing research in this field made one or more of the impractical assumptions, that the information about the spurious correlations is explicitly known to models during training (e.g., knowing that age is the bias attribute and having access to objects' ages in gender classification), that the spuriously correlated attributes are binary (yes or no, existent or non-existent, etc.), that only one spuriously correlated attribute exists, etc. These assumptions hinder applying proposed methods to a broader scope.

2.1 Re-weighting

Re-weighting methods give more weight to samples in minority groups. When there is only one binary biased attribute, the minority group is indeed the discriminated group, as models learn to map the dominant biased pattern to labels. Therefore, forcing models to learn more from the less-represented group diminishes models' exploitation of the bias attribute. However, when the number of bias attributes increases or when bias attributes can not be parameterized, it is challenging to assign a value to each sample to measure the level of bias of each sample.

Cui et al. [10] approached the problem by introducing a re-weighting formula. The concept of "effective number of samples" was defined to estimate the expected volume of feature spaces covered by samples, and was used to derive a class-wise re-weighting formula controlled by a hyper-parameter. The hyper-parameter controlled the impact of re-weighting from no re-weighting to re-weighting by inverse class frequency, while classes with fewer samples would get higher weights.

Ahn et al. [11] suggested a two-step training. First, a biased model was trained with the original biased dataset, after which every sample would have a gradient norm. Second, a smaller training set was re-sampled from the original dataset, while the probability of choosing any sample was proportional to its gradient norm obtained from the first step. Given the observation that rare samples had larger gradient norms in the first step, the re-sampling operation generated a better-balanced training set, as rare samples were more likely to be in the new training set.

Similarly, Chai and Wang [12] proposed a method to assign adaptive weights to samples during the training. Positive weights were only assigned to samples that were likely to be misclassified, thus forcing the models to learn more from the minority samples with under-represented features.

2.2 Modifying Loss Function

As loss functions direct the learning goal of models, modifying loss functions to depreciate spurious correlations undoubtedly turns models blind to bias. However, altering loss functions makes assumptions about what bias attributes are like. Previous modifications assumed that the biased attributes were binary, and some methods needed information about which attributes were biased.

Many tried to solve the problem as a Distributionally Robust Optimization (DRO) problem [13, 14, 15]. DRO guesses the test distribution by sampling from the training set and minimizes the worst-case expected loss based on the potential test distribution. Among them, Sagawa et al. [15] argued that naive DRO approaches, while trying to vanish training loss, still achieved low accuracy on worst groups. They found that instead of focusing on lowering general training loss, selectively lowering training loss on the groups with larger expected generalization gaps helped to improve worst group accuracy. However, specifying the bias was necessary to apply this method.

Ahmed et al. [16] provided another approach. Introducing the distance between group distributions as a regularization term into the loss function encouraged the model to favor similar class-conditioned distributions of features, thus the model was prompted to learn group-invariant features. However, such adjustment made assumptions that bias attributes were binary.

2.3 Adversarial Learning

Adversarial learning approaches followed the idea of learning the biases to unlearn them. As promising as it sounds, these methods require information about the biased attributes to set the learning goal of the bias-learning components, but such information is not always obtainable.

Kim et al. [17] introduced an additional deep neural network into the model. Aside from the main network that was responsible to make classifications, another deep neural network was added to predict the bias labels from samples. Besides, the main classifier was trained to minimize the negative conditional entropy regarding the bias predictor, thus the classification was made with fewer dependencies on bias attributes. However, training the bias predictor required bias labels.

Adding to that, Majumdar et al. [18] introduced an attention mechanism. The model had two branches, namely main classifier and bias predictor. The main classifier and the bias predictor both had an attention module attached after their average pooling layers, while the copy of the attention module of the bias predictor was reversed and put into the main classifier to augment its original attention module. Therefore, the main classification network made predictions based more on unbiased regions, as the attention module of the bias predictor emphasized the biased regions. However, this improvement did not eliminate the requirement for bias labels.

Alvi et al. [3] attached multiple classifiers to the latent layer. The main classifier predicted the labels of the samples, while other auxiliary classifiers learned confusion loss of spurious variations. Then these bias-learning classifiers were encouraged to alter the latent layer such that the feature representation contained less information about the spurious features, therefore the main classifier could learn from unbiased feature representation. Again, training the model required labels of spurious variations from datasets.

2.4 Popular Biased Image Classification Datasets

Many researchers modified the MNIST dataset for their work [16, 11, 19, 20, 21, 22, 23, 9, 24]. Colors were added to the MNIST dataset as the spurious correlation. Other popular datasets included CelebA, where spurious correlations were hair colors and gender [18, 5, 9]. However, these two datasets have a too simple bias to test the performance of bias mitigation methods facing more complicated bias, for example, the underlying bias among images in popular datasets like ImageNet and CIFAR, so some works also synthesized their biased datasets [3, 25, 20].

3 Solution

While the attention mechanism proposed by Majumdar et al. [18] could address multiple non-binary biases, it required bias labels from datasets. Cui et al. [10] noted that biased samples and unbiased samples behaved differently in a model. By experiment, the vectors after the fully-connected layer showed a difference (details in Section 4.3). This feature was utilized to differentiate samples where bias labels were not available. Feeding the differentiated samples into the attention-mechanism-attached model trained the attention module to detect the biased regions. By using the reversed attention module, the model learned not to focus on the biased regions and became a fair model from a biased dataset without specified bias labels.

3.1 Objective

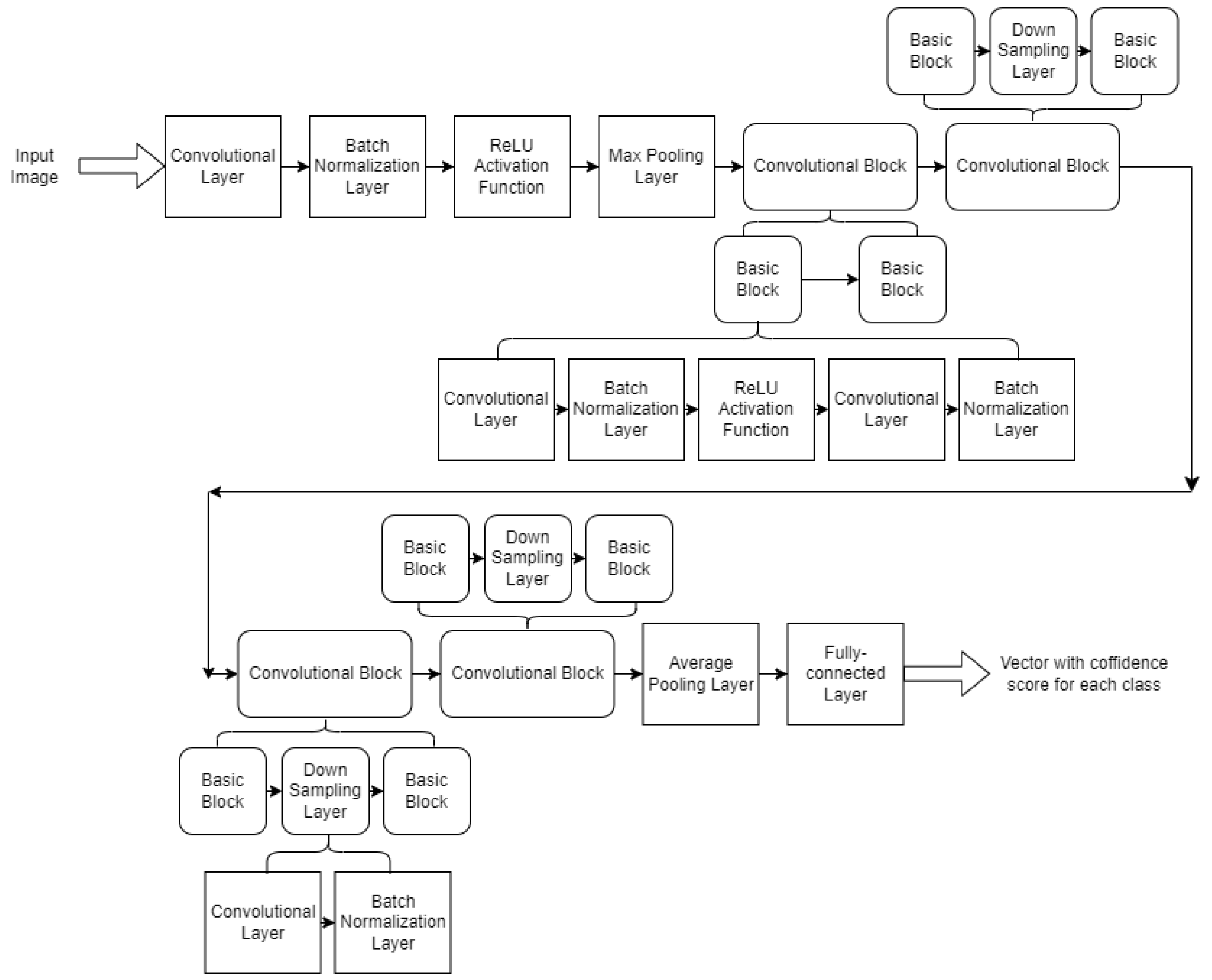
Suppose there is an image set $\mathcal{X} = \{x_i\}_{i=1}^{m}$ and a label set $\mathcal{Y} = \{y_j\}_{j=1}^{n}$, where $x_i \in \mathcal{X}$ is an image and $y_j \in \mathcal{Y}$ is a label. Suppose each image has a corresponding label. Suppose that there are K spurious variables and that for each sample x_i, there exists an unobserved $s_i \sim \mathcal{S}$ where $\mathcal{S}$ is a random variable over $\{k\}_{p=1}^{K}$. Suppose that for each sample x_i, there exists a $t_i \in \mathcal{T}$ where

$\mathcal{T}$ is the true patterns to be recognized. I.e., there exists a one-to-one function that maps $\mathcal{T}$ to $\mathcal{Y}$. To summarize, there are m (x_i, s_i, t_i, y_j) pairs. Suppose that there are no causal relationships between t_i and s_i. Suppose that in the training set, among the image samples belonging to each y_j, more than half of the samples share the same s, while in the testing set, among the image samples belonging to each y_j, no more than half of the samples share the same s. The objective of this project is to train an image classification model using the training set and test its performance using the testing set without knowing $s, k, \mathcal{S}, K$. The classification accuracy and recall of the modified model should be higher than those of the baseline model.

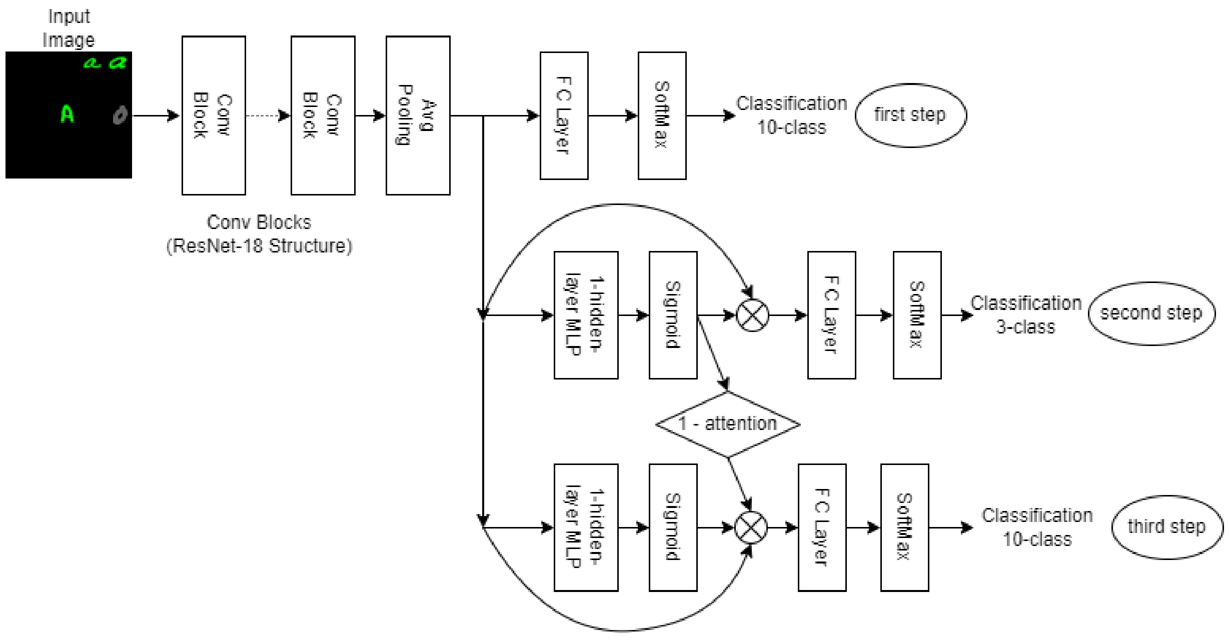
3.2 Model Structure

He et al. [26] proposed the ResNet architecture, where models had similar structures and the difference was the number of convolutional blocks and the size of convolutional layers. The ResNet-18 was chosen to be the baseline model. The ResNet-18 structure started with a convolutional layer, a batch normalization layer, a ReLU activation function, and a max pooling layer. After that, it had four convolutional blocks, where each block shared similar architectures. Each convolutional block had two basic blocks, while each basic block had a convolutional layer, a batch normalization layer, a ReLU activation function, another convolutional layer, and another batch normalization layer, from start to end. In the last three convolutional blocks, there is an additional downsampling layer, which is a convolutional layer followed by a batch normalization layer, between the two basic blocks. After the four convolutional blocks, ResNet-18 had an average pooling layer and a fully-connected layer. Figure 1a illustrates its structure.

A multi-task adversarial learning model was proposed. The model had three branches after the average pooling layer, with the three branches sharing the same convolutional blocks and average pooling layer structure as the ResNet-18. All three branches had a fully-connected layer followed by a softmax activation function. In the second and third branches, before the fully-connected layer, there was a one-hidden-layer multi-layer perceptron followed by a sigmoid function. They served as an attention module. There was a connection between the average pooling layers and the fully-connected layers in the second and third branches, where the output of the average pooling layers and the multi-layer perceptrons performed element-wise multiplication before entering the fully-connected layers. Additionally, in the third branch, one minus the output vector of the sigmoid function of the second branch also joined the element-wise multiplicatron before the fully-connected layer. Figure 1b shows the structure of the proposed model.



(a) ResNet-18 Model



(b) Proposed Model

Figure 1: Model Structure

3.3 Training Procedure

The training process involved three steps.

First, the first branch was trained using a vanilla training set from the dataset. After training, the training set was re-input into the first branch and the outputs of the fully-connected layer were collected. The highest values in these vectors were extracted and normalized to have a mean of 0 and a variance of 1.

Second, the second branch was trained using training images from the dataset, while their labels were changed to their corresponding normalized values. The second branch learned to classify if a sample's value was above α, below $-\alpha$, or between $-\alpha$ and α, where α was a hyper-parameter.

Third, the third branch was trained using a vanilla training set from the dataset. After training, it was used to make final predictions of samples input into the model.

A pseudocode is provided in Listing 1:

```
initialize model1 for the first step
model1.train(training_segment)
model1.freeze()

confidence = model1.predict(training_segment).output_FC
new_segemnt = {training_seg_sample: compare(sample's confidence,threshold)}

initialize model2 for the second step
model2.inherit(model1's convolutional blocks)
model2.train(new_segment)
model2.freeze()

initialize model3 for the third step
model3.inherit(model1's convolutional blocks)
configure model3's attention modeul using model2's attention module
model3.train(training_segment)
```

Listing 1: pseudocode for training procedure

4 Results and Discussion

4.1 Experimentation protocol

The model was implemented using the PyTorch package in python and was trained using PyTorch Lightning. The code was available on GitHub[1]. The model was trained on NYU Shanghai HPC,

[1]GitHub: https://github.com/Ruochen1105/Capstone

with 12 g6230 CPUs and 8 GeForce RTX 2080 Ti GPUs. The first, second, and third branch was trained for 90, 40, and 90 epochs respectively. In the second branch, the hyper-parameter α was set to be 0.5. Other hyper-parameters were set to be the same as the original ResNet-18 [26].

4.2 Dataset Used

BiasedMNIST dataset proposed by Shrestha et al. [9, 27] was used in the experiment. The BiasedMNSIT dataset was synthesized from the MNIST dataset by introducing digit color, digit positions, colored letters, and colored textures as distractors. A hyper-parameter controlled, for each distractor, the proportion of samples in a class that shared the same distractor. The correlation level of 0.9 was chosen in the experiments, so 90% of the samples of the same class would share the same digit color, and another 90% would share the same digit position, etc.

4.3 Detecting if Samples are Biased

After the first branch was trained for 90 epochs, the model was frozen. The training samples were again input into the model, and the vectors before the activation layer were extracted. The highest values of these vectors were picked and grouped based on the label of the samples. Within the same group, these picked values were normalized with a mean of 0 and a variance of 1. The visual inspection of the outcome suggested that, within the same class, a classifier trained using biased data will assign higher confidence scores to biased samples compared to unbiased samples. Table 1 demonstrated some training samples from the "0" class and their normalized highest values from the extracted vector. It could be visually recognized that biased samples, or samples with a grey "0", two green "a," and a green "A" were given positive scores, while samples with random distractors were given negative scores. Please note that 0 was not necessarily separating biased and unbiased samples and the threshold that best classified if a sample was biased needed parameter-searching and might be dataset-specific. However, as proved by the experiment result described in Section 4.4, even though the exact threshold was unknown, the property that biased samples tended to have higher fully-connected layer output values helped to mitigate the bias.

4.4 Bias Mitigation Effect

To understand if the proposed modification helped to mitigate bias from datasets, after the completion of the first step and the third step, the first and third branch were used to predict the labels of samples from the testing segment respectively. The experiment was conducted for

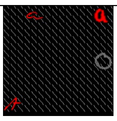
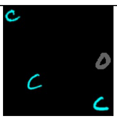
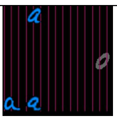
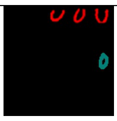
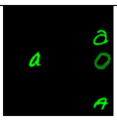
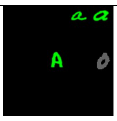
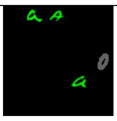
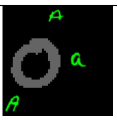
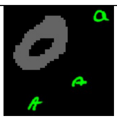
unbiased	 -0.73	-1.06	 -1.38	 -1.48	 -2.04
biased	 0.16	 0.74	 1.02	 1.17	 1.28

Table 1: some training samples from the "0" class and their normalized highest values from the extracted vector

class	unmodified ResNet-18	modified ResNet-18	improvement
0	0.522 (0.000379)	0.617 (0.000473)	0.095 (18.2%)
1	0.771 (0.000446)	0.794 (0.000264)	0.023 (2.98%)
2	0.474 (0.000320)	0.525 (0.0000990)	0.051 (10.8%)
3	0.489 (0.000235)	0.531 (0.000517)	0.042 (8.59%)
4	0.645 (0.000449)	0.675 (0.000155)	0.030 (4.65%)
5	0.560 (0.00144)	0.641 (0.000416)	0.081 (14.5%)
6	0.685 (0.000353)	0.707 (0.000171)	0.022 (3.21%)
7	0.630 (0.000310)	0.667 (0.000533)	0.037 (5.87%)
8	0.447 (0.000473)	0.479 (0.000369)	0.032 (7.16%)
9	0.457 (0.000426)	0.504 (0.000688)	0.047 (10.3%)
overall	0.572 (0.00184)	0.616 (0.00000659)	0.044 (7.69%)

Table 2: statics of f-1 scores

seven times. The mean and variance of the class-wise and overall testing f-1 scores were shown in Table 2. Overall and for each class, the proposed model increased the testing f-1 score. As f-1 score measured the "correctness" of the model's decision, a higher f-1 score suggested that the proposed model's performance on the testing set was better than that of the baseline model. Some corrected samples were illustrated in Table 3. Therefore, the proposed modification on the baseline model did help to mitigate dataset bias.

5 Discussion

Compared with previously proposed methods, the modification proposed in this project did not have any requirement on any information other than the training image and labels, thus the modification could be applied to more datasets. Besides, as the modification only added attention modules before the fully-connected layer and left other parts untouched, it could be combined with more modules. In general, the project proved that it was possible to utilize the property that biased and unbiased samples performed differently in models to design bias mitigation methods.

Due to the limitation of time and computational resources, there was no hyper-parameter

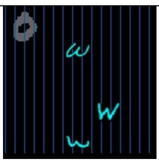
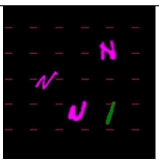
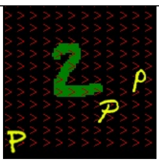
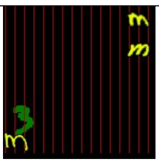
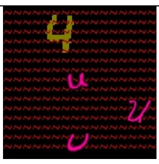
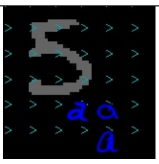
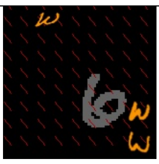
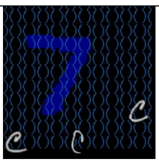
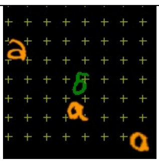
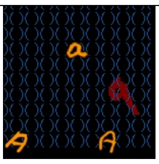
sample	wrong	corrected	sample	wrong	corrected
	6	0		7	1
	9	2		9	3
	8	4		1	5
	0	6		5	7
	0	8		3	9

Table 3: misclassified samples being corrected by the proposed model

searching in this project. Increasing the size of the hidden layer in the one-hidden-layer multi-layer perceptron could increase its capacity to store information and affect the model, which could potentially boost the performance. Besides, the hyper-parameter $\alpha = 0.5$ in the second branch was decided arbitrarily, and how to set α was worth exploring.

During training, model's performance dropped sharply if any step of it was trained for fewer epochs. The model might have utilized overfit network weights to better locate bias patterns, which could be a future research direction. Besides, the improvement was not even among classes and was not related to classes' original f-1 scores. Whether the model was exploiting certain characteristics of the biases introduced in the BiasedMNIST dataset was worth researching.

Also, stronger attention mechanisms like CBAM [28] may enable the model to better focus on the bias regions, thus increasing the performance of the model.

Besides, the project only trained and tested using the BiasedMNIST dataset with a correlation level of 0.9. Future research should train and test the model using various datasets to make sure that the method is not exploiting how BiasedMNIST is introducing biases. Also, the performance of other models should be tested to compare with the performance of the proposed model.

6 Conclusion

This project proposed a bias mitigation model that, unlike the previous methods, did not require bias information from datasets. The experiments proved that the model did increase the testing f-1 score for models trained on a biased training set but tested on an unbiased testing set.

This project suggested that the feature that biased samples had higher fully-connected layer output values than unbiased samples could be used to design debiasing methods without bias labels available.

References

[1] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[2] K. Kafle, R. Shrestha, and C. Kanan, "Challenges and prospects in vision and language research," *Frontiers in Artificial Intelligence*, vol. 2, p. 28, 2019.

[3] M. S. Alvi, A. Zisserman, and C. Nellåker, "Turning a blind eye: Explicit removal of biases and variation from deep neural network embeddings," *ArXiv*, vol. abs/1809.02169, 2018.

[4] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[5] M. Pezeshki, O. Kaba, Y. Bengio, A. C. Courville, D. Precup, and G. Lajoie, "Gradient starvation: A learning proclivity in neural networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1256–1272, 2021.

[6] J. Attenberg, P. Ipeirotis, and F. Provost, "Beat the machine: Challenging humans to find a predictive model's "unknown unknowns"," *Journal of Data and Information Quality (JDIQ)*, vol. 6, no. 1, pp. 1–17, 2015.

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

[8] M. Hort, Z. Chen, J. M. Zhang, F. Sarro, and M. Harman, "Bia mitigation for machine learning classifiers: A comprehensive survey," *arXiv preprint arXiv:2207.07068*, 2022.

[9] R. Shrestha, K. Kafle, and C. Kanan, "An investigation of critical issues in bias mitigation techniques," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1943–1954.

[10] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9268–9277.

[11] S. Ahn, S. Kim, and S.-y. Yun, "Mitigating dataset bias by using per-sample gradient," *arXiv preprint arXiv:2205.15704*, 2022.

[12] J. Chai and X. Wang, "Fairness with adaptive weights," in *International Conference on Machine Learning.* PMLR, 2022, pp. 2853–2866.

[13] J. C. Duchi, T. Hashimoto, and H. Namkoong, "Distributionally robust losses against mixture covariate shifts," *Under review*, vol. 2, 2019.

[14] H. Rahimian and S. Mehrotra, "Distributionally robust optimization: A review," *arXiv preprint arXiv:1908.05659*, 2019.

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

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

[17] B. Kim, H. Kim, K. Kim, S. Kim, and J. Kim, "Learning not to learn: Training deep neural networks with biased data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9012–9020.

[18] P. Majumdar, R. Singh, and M. Vatsa, "Attention aware debiasing for unbiased model prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 4133–4141.

[19] Y. Hong and E. Yang, "Unbiased classification through bias-contrastive and bias-balanced learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 26 449–26 461, 2021.

[20] E. Kim, J. Lee, and J. Choo, "Biaswap: Removing dataset bias with bias-tailored swapping augmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 14 992–15 001.

[21] J. Lee, J. Lee, S. Jung, and J. Choo, "Debiasbench: Benchmark for fair comparison of debiasing in image classification," *arXiv preprint arXiv:2206.03680*, 2022.

[22] Z. Li, A. Hoogs, and C. Xu, "Discover and mitigate unknown biases with debiasing alternate networks," *arXiv preprint arXiv:2207.10077*, 2022.

[23] Y. Qiang, C. Li, M. Brocanelli, and D. Zhu, "Counterfactual interpolation augmentation (cia): A unified approach to enhance fairness and explainability of dnn," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22, LD Raedt, Ed. International Joint Conferences on Artificial Intelligence Organization*, vol. 7, 2022, pp. 732–739.

[24] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy, "Domain generalization: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[25] M. Jeon, D. Kim, W. Lee, M. Kang, and J. Lee, "A conservative approach for unbiased learning on unknown biases," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 752–16 760.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] R. Shrestha, K. Kafle, and C. Kanan, "Occamnets: Mitigating dataset bias by favoring simpler hypotheses," *arXiv preprint arXiv:2204.02426*, 2022.

[28] S. Woo, J. Park, J.-Y. Lee, and I.-S. Kweon, "Cbam: Convolutional block attention module," in *European Conference on Computer Vision*, 2018.