Rectifying Shortcut Learning via Cellular Differentiation in Deep Learning Neurons

Hongjing Niu
sasori@mail.ustc.edu.cn
Guoping Wu
guoping@mail.ustc.edu.cn
Hanting Li
ab828658@mail.ustc.edu.cn
Bin Li
binli@ustc.edu.cn
Feng Zhao ✉
fzhao956@ustc.edu.cn

MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition, University of Science and Technology of China, Hefei, China

Abstract

Deep learning models have exhibited tendencies to rely on shortcuts, such as identifying sheep based solely on the presence of grassland. This reliance often overshadows the true potential of deep learning, significantly impacting their performance and robustness in diverse scenarios. Particularly, shortcuts embedded in the training data may prove detrimental in the testing scenarios, acting as sources of interference rather than assistance. A notable challenge arises in biased datasets, where the learning of the intended attributes is hindered by spurious correlations with extraneous attributes. This paper introduces an innovative method that harnesses positive feedback to foster internal differentiation within the model, thereby diminishing feature entanglement and enhancing the learning of distinct, non-shortcut features. Our approach employs model parameter masks to segment the model into locally specialized subcomponents. This segmentation facilitates the amplification of differences between these components through interaction, allowing for the distribution of the model's overall functionality across each subcomponent. In essence, our method enables the independent learning of diverse features, circumventing the model's reliance on shortcut features. Comprehensive experiments conducted on a variety of datasets from multiple perspectives have demonstrated the efficacy of our proposed method. It significantly enriches the feature diversity and improves the model performance in complex scenarios.

1 Introduction

Despite the remarkable achievements in various fields [8, 12, 17], deep learning models still suffer from shortcut learning [7]. Shortcut learning refers to the adoption of decision rules that are effective to some extent but differ from the designer's intended goals. We refer to

✉ Corresponding author.

the features adopted by such decision rules as shortcuts. For example, a model designed to identify "sheep" may rely on the background "grassland" as a decision criterion. In such an example, the target features are directly related to sheep, while features related to grasslands are shortcuts. So it is not hard to understand shortcut may not only diminishes the model's generalization performance but also affects its reliability.

Research on shortcuts focuses on how to exclude shortcuts that are known to some extent, which raises some questions. For example, what features could potentially be shortcuts, which may not be known to us; features identified as shortcuts may not necessarily be detrimental to the task; the process of excluding shortcuts inevitably compromises the modeling of target features. We will conduct a more specific analysis in Section 2.

Unlike them, this paper focuses on learning multiple distinct features rather than rectifying some of them directly. Our approach disentangles features and enhances the individual learning of them. To disentangle diverse features and learn them adequately, we propose a method to promote "internal differentiation" in the model. Our method identifies a set of sub-models within a single model, each with a differentiated focus on representing different features. The final prediction of the model is based on the collective decision of sub-models.

The contributions of this paper can be summarized in the following points:

- The paper proposes a method for feature differentiation in models, which can learn more disentangled feature representations and enables the model to better adapt to complex task scenarios.
- The proposed method offers a new strategy to avoid excessive reliance on shortcut, which no longer requires prior knowledge of shortcut features.
- This work provides a new approach for debiasing tasks and achieves excellent performance without introducing additional priors or assumptions.

2 Related Work

The idea of inhibiting shortcuts has been widely applied in various research scenarios. We divide those works based on the prior knowledge used and have a brief discussion.

Sometimes, we know what features can be shortcuts based on the analysis of the scene. For example, the suppression of camera features is employed in person re-identification [5], and identity information is excluded in facial expression recognition [30]. Once the shortcuts to be inhibited are clearly identified, two main technical approaches can be employed, data augmentation [10, 19, 29] and regularization [3, 9, 26].

Sometimes we know very little about shortcuts, which makes the task more difficult. At this point, some certain prior knowledge or assumptions can also be helpful. Some works suppress the learning of specific features such as texture [2, 28], which could be inspired by the learning of texture preference of convolutional neural networks [6]. Some assumptions are also used, such as the idea that shortcuts are easy to learn [4, 23, 25]; those samples that have a higher gradient during training are the ones where shortcuts do not hold, and they should receive more training [1]. Many adversarial training methods [20, 21, 22, 24] or augmentation [10, 31] methods are also based on these assumptions.

In general, most methods require more relevant prior knowledge or assumptions, which limited their applicability. We aim for a method with fewer preconditions.

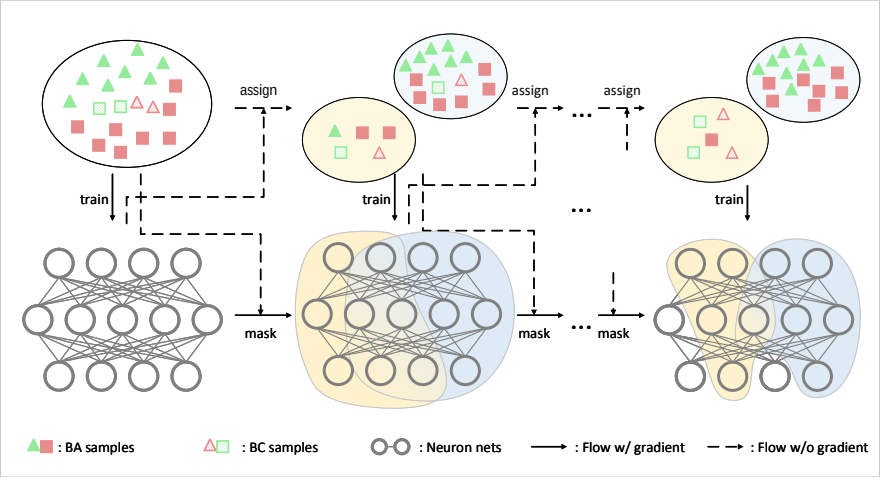


Figure 1: Schematic of the framework. The dataset and neural network change from left to right as the basic training cycle is repeated. The neural network is represented by neurons and their connections, and the yellow and blue regions in the neural network are indicated to perform the corresponding masking. We also use the colors to indicate the reassigned dataset in relation to the corresponding masked model.

3 Methodology

3.1 Problem Definition

We follow the idea of shortcut learning[7] and present a formalized definition in a supervised classification task. Let $(X,Y) \in (\mathcal{X},\mathcal{Y})$ refer to a sample and the corresponding label. Each sample have M independent attribute, denoted as $(A_1,A_2,...,A_M)$. We assume that only one of these attributes, A_t, is the target attribute we wish to learn. In the training set D^{train}, whether an attribute A_m is helpful for the classification can be reflect by $I(A_m;Y)$, the mutual information between A_m and Y. In particular, A_m and Y are independent of each other when $I(A_m;Y) = 0$, and when $I(A_m;Y) = H(Y)$, knowing A_m is theoretically sufficient to predict Y correctly. $H(Y)$ refers to the information entropy of Y. We perform a normalization and use predictive power $PP(A_m;Y) = I(A_m;Y)/H(Y)$ to measure A_m's ability to predict Y. Note that the dataset will affect the distribution of A and Y, and therefore the calculation of $PP()$. The use of $PP()$ gives the dataset and omits Y.

$$PP^D(A_m) = I(A_m^D;Y^D)/H(Y^D) \tag{1}$$

In a testing set D^{test}, the distribution of data might be different from that in D^{train}, which means the mutual information between A_m and Y could be changed. In such an environment, the only certainty is that the target attribute A_t remains perfectly predictive of the label Y, while the relations between some other attribute and label are changed:

$$\begin{aligned} &PP^{train}(A_t) = PP^{test}(A_t) = 1; \\ &PP^{train}(A_m) \neq PP^{test}(A_m), m \neq t. \end{aligned} \tag{2}$$

With such definitions, shortcut attribute A_s can be described as the attribute that the predictive power decreases from the training set to the testing set. Thus, shortcut learning

indicates that the model M learns the shortcut attribute and prefers it rather than the target attribute for prediction, while the predictive power of shortcut attribute is less than 1 (otherwise it should be the target).

$$I(M(X);A_s) > I(M(X);A_t); PP^{train}(A_s) > 1-\varepsilon. \tag{3}$$

where ε is a small positive value. So the model might learn A_s. The ε reflects the performance that can be achieved solely by A_s.

For the sake of narrative and experimental simplicity, we only consider the target attribute A_t and one shortcut attribute A_s and assume that they have the same value ranges, formulated as $|\mathcal{A}_t| = |\mathcal{A}_s| = |\mathcal{Y}|$. Note that real-world scenarios may have multiple attributes, but dividing them into two groups is sufficient for most scenarios, as A_s and A_t can both be attributes with composite features. And the promotion of this method is not complicated, especially compared to other related works we have compared. We also conducted experiments on multi-attribute data. The correspondence that exists between A_t and A_s in the majority of the training samples is represented by the same values of A_t and A_s. Thus, we call the sample where A_t and A_s take the same value the bias-aligned (BA) sample and the sample that takes different values the bias-conflicting (BC) sample. To validate the performance of the model, a natural approach is to test the model's performance on an testing set where $I(A_s;Y) = 0$. For a better view of the results, we also look at the performance on the BC samples.

3.2 Neuron Differentiation

As mentioned earlier, there is a target attribute A_t and a shortcut attribute A_s in the experimental scenario of the debiasing task. And an ideal model should have a good representation of the target attribute. However, A_s has high predictive power in the training set, thus model can hardly ignore A_s and focus on learning A_t. A feasible strategy, without introducing additional priors, is to represent attributes with high predictive power separately. By introducing the idea of cell differentiation in deep neural networks, our approach promotes neural networks to learn about different attributes and represent them with different sets of neurons.

The overall framework is shown in Figure 1. The training process starts with a dataset D and a neural network for classification $M_\theta : \mathcal{X} \longrightarrow \mathcal{Y}$, where $\theta \in \Theta$ represents for the learnable parameters in M. Note that there is only one dataset and one model all the time. From left to right in the figure are different divisions of the dataset, and models with different training phases and masks. We first optimize M so that it has some representational power over D. We use generalized cross entropy (GCE) [32] as a loss to amplify the model's reliance on shortcuts, in order to facilitate faster subsequent positive feedback processes.

$$L_{GCE}(x,y;M) = \frac{1-p(M(x),y)^q}{q}, \tag{4}$$

where p denotes the probability that M predicts sample x for category y, and $q \in (0,1]$ is a hyperparameter that regulates the loss, with smaller q meaning that the loss is closer to cross-entropy and larger q meaning that the loss is closer to mean absolute error. The model is used to evaluate the training set and the samples that can be correctly classified and those that cannot are divided into two subsets named D_a and D_b. D_a is mainly composed of incorrectly categorized samples and a small number of correctly categorized samples, whereas D_b consists of correctly categorized samples. Next, the method will look for sub-models in the model that best fit each of the two subsets. The search for best sub-models is achieved

by learning the best-performing mask. To ensure that the gradient is available for training and enhance the exploitation of the method, Gumbel softmax trick [11] is utilized for mask generation. In other words, masks $m_a, m_b \sim G(z)$ denotes the sub-model for learning D_a, D_b. Optimizing mask for D_a and D_b can be translated into reducing loss L_m as follows:

$$L_m = L_s(m_a, D_a) + L_s(m_b, D_b) + IoU(m_a, m_b). \tag{5}$$

In Equation, Intersection-over-Union (IoU) between m_a and m_b is used to constrain the two masks to have as few common regions as possible. Loss of a sub-model L_s calculate the GCE loss and count the size of the mask as a regularity term thus shrinking the mask.

$$L_s = GCE(x, y, M \circ m_a) + \lambda \sum_{m_a} |\theta_i|. \tag{6}$$

For the latter steps, the division of D_a and D_b is according to the correctness of prediction by sub-model. The correctly predicted samples will be retained in the sub-dataset, while the incorrectly predicted samples will be assigned to another sub-dataset. In this way, a sub-model learning more on the samples that are easy for itself, while the difficult samples are passed on to other sub-models for learning. Over time, the feature representations between the sub-models gradually become differentiated.

When the evaluating process is executed, mask will be converted from the sampling result to the form 0-1. Therefore, some fine-tuning on the training data was required before the evaluating process(re-assigning training set and testing on testing set). In summary, our method can be written as the following pseudo-code in Algorithm 1.

Algorithm 1 Neuron Differentiation

Require: Dataset D, model M with parameter Θ.
Ensure: Trained model parameter Θ^*, masks $\mathbf{m}$.
  Initialize Θ as Θ_0.
  Update Θ with L_{GCE} in Eq. 4.
  Split and assign D into D_a and D_b.
  Initialize masks m_a and m_b.
  repeat
    Update m_a and m_b with L_m in Eq. 5.
    Fine-tuning Θ for evalutation.
    Evaluate D on $M \circ m_a$ and $M \circ m_b$.
    Re-assign D_a and D_b.
  until Re-assigned D is consistent the previous.

4 Experiments

4.1 Setups

Datasets To better explore the performance of our method on biased datasets, we conducted experiments on multiple biased datasets, both synthetic (CMNIST [18],CIFAR10-C [18]) and natural (BFFHQ [18], BAR [23]). We provide a brief description of these datasets to facilitate a better understanding of the experiments. CMNIST is synthesized by coloring

the digits in MNIST [16]. So the target attribute and shortcut attribute A_t, A_s are digit and color. CIFAR10-C is another classical synthesized dataset for debiasing task. The samples are generated by adding corruptions to the CIFAR-10 [15] samples (A_t, A_s are object and corruption). BFFHQ is a widely used real-world dataset for debiasing, which is created from the FFHQ dataset [13]. It contains human face images annotated with their facial attributes, where A_t and A_s are gender and age. BAR is another real-world dataset for debiasing in which A_t and A_s are action and background. For the synthetic dataset, we can easily control the ratio of the percentage of bias-conflicting (BC) samples, which is 0.5% for BFFHQ. For BAR, there are no explicit BC samples in the training set.

Implementation details Our code is included in the "code" document in supplementary materials and will be open sourced. Our experiments were all run on RTX-4090, and the code was implemented based on the PyTorch framework. We use a simple convolutional neural networks with 3 conv-layer (kernel nums are 64, 128, 256) and 3 batch-normalization layer for experiments on CMNIST, and ResNet-18 for others (pre-trained parameters provided by torchvision are used). The batch size are set as 256 for CMNIST and CIFAR10-C, 64 for BFFHQ.

For CMNIST, we first pre-trained model for 2k iterations, we used GCE loss (0.7) and SGD as the optimizer (learning rate=0.01). Then the model parameters were re-initialized and trained for 10k iterations, CE was used and the optimizer is Adam (learning rate=0.01). Then for the differentiation, we trained the mask for 2k iterations with Adam (learning rate=0.01) and the coefficients of regularized terms λ was set to 1e-8. The data assignment was updated every 3 epoch, according to the result of sub-model A & B.

For CIFAR10-C and BFFHQ, the pre-trained parameters provided by torchvision was used. We train 10k iterations with Adam (learning rate=1e-3). When training to find the mask, the λ was set to 1e-9, because the features are more complicated, and a slower training rhythm is required. We conducted epoch-ensemble-based mining algorithms as that in [25]. The B-C score threshold τ was set as 0.8 for CIFAR10-C and 0.7 for BFFHQ. The confidence threshold was 0.05 for both datasets.

Dataset re-assignment was conducted by combining wrong-answered data and a little random data. For sub-model A, the wrong-answered data has the weight of $(CategoryNum - 1)/WrongAnsweredRatio$. For sub-model B, the wrong-answered data has the weight of 0.

4.2 Debiasing on Different Attributes

We first tested the proposed method's ability to mitigate bias across different datasets. As mentioned earlier, these datasets have different combinations of attributes, making them suitable for demonstrating the method's generalizability. Some of the methods mentioned in baseline are also compared. The results of our experiments are all the mean value of at least three randomizations. Some standard deviations are omitted due to space limitations.

The results are shown in Table 4.2, and our method outperforms the other methods being compared. Further, as the percentage of BC samples decreases and the difficulty of the task increases along with the dataset bias, our method has a more significant improvement.

4.3 Differentiation of Sub-models

Unlike previous methods intended to learn an unbiased model, we wish to learn different attributes in the dataset with different sub-models. In order to verify that the effectiveness of our approach indeed benefits from the realization of our original design intention, we

Table 1: Image classification accuracy biased training datasets. The ratio denotes the percentage of bias-conflicting (BC) samples in the training set. A lower ratio means greater data bias, i.e., higher task difficulty. A check mark (✓) indicates that no priors about the feature are used, while a cross mark (×) means the opposite.

Dataset	Ratio (%)	Vanilla‡	EnD‡	ReBias‡	LFF‡	DFA‡	DCWP‡	Ours
		✓	×	×	✓	✓	✓	✓
CMNIST	0.5	62.36	84.32	69.12	83.73	86.74	93.41	**93.68**$_{\pm 0.47}$
	1.0	81.73	94.98	84.65	88.44	93.15	95.98	**96.15**$_{\pm 0.16}$
	2.0	89.33	97.01	91.96	92.67	95.15	97.16	**97.23**$_{\pm 0.16}$
	5.0	95.22	98.00	96.74	94.90	96.76	**98.02**	97.86$_{\pm 0.04}$
CIFAR10-C	0.5	22.02	23.93	21.73	27.02	27.86	35.90	**38.60**$_{\pm 0.39}$
	1.0	28.00	27.61	28.09	31.44	34.62	41.56	**45.53**$_{\pm 1.58}$
	2.0	34.63	36.62	35.57	38.49	41.95	49.01	**51.78**$_{\pm 0.61}$
	5.0	45.66	43.67	48.22	46.16	49.15	56.17	**59.04**$_{\pm 0.90}$
BFFHQ	0.5	52.25	59.80	54.90	56.50	55.50	60.35	**60.70**$_{\pm 1.30}$

visualized the features extracted by the models using t-SNE [27], and the results are shown in Fig 2. We visualized the two differentiated sub-models (column 1 & 3) separately, and a vanilla model trained on the dataset was used as a control set. The models are trained on CMNIST with 0.5 ratio. The appendix contains more experimental results.

The same row represents the features extracted by models for the same set of testing data. In the top row, we use different colors to indicate the values of target attribute. In the bottom row, different colors indicate the values of bias attribute. The label of bias attribute is only used for visualization and is not available during training.

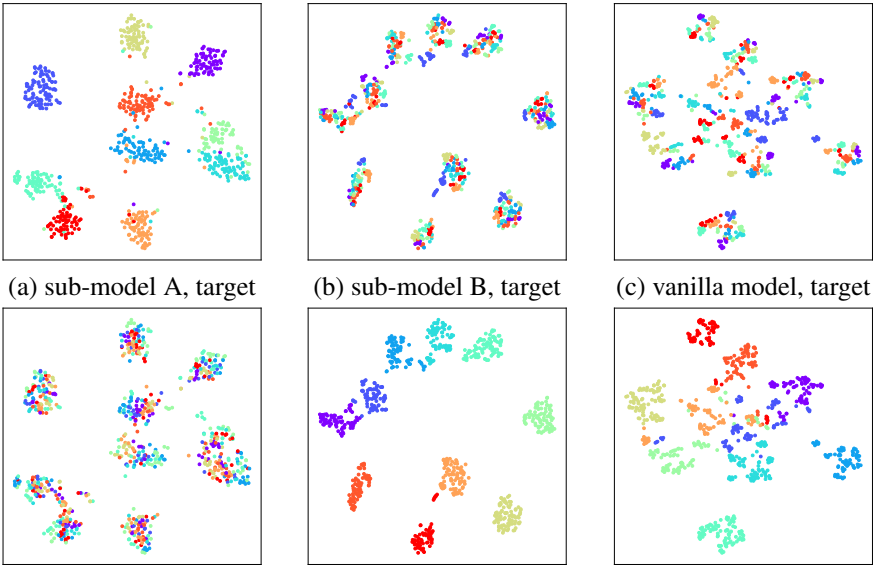
From (c) and (f) we can see that the target attribute features can hardly be distinguished while the distinguish of shortcut attribute is much easier. Such observation indicates that the model learns the shortcut attributes better, while the learning of target attributes is poor. Let's focus on (a) and (d). It can be observed that the sub-model A has diametrically opposed performance to the vanilla model. Sub-model A can distinguish target attribute well but fail to tell shortcut attribute. On the other side, it can be seen from (b) and (e) that sub-model B has good discriminatory properties for the shortcut attribute, but confuses the target attribute. Compared to the vanilla model, sub-model B has a purer learning of bias attributes.

Combining the above results, we can draw the following observations:

- The vanilla model prefers shortcut attribute and also learns target attribute. The model's representation of features is a mixture of the both attribute features.
- Under the action of the proposed differentiation method, the model's mixed representation of the attributes is separated into different sub-models.Each sub-model learns only one single attribute and has good invariance to the other attributes.

To further explore the association between the feature representations of the sub-models, we analyzed each layer of the sub-models using centered kernel alignment (CKA) [14]. CKA is an analytical tool to understand the behavior of neural networks by comparing representations between layers and between different trained models. The higher CKA value somewhat reflects the higher similarity between the two layers.

We tested the CKA values between the layers of the two sub-models and the vanilla model in both CMNIST and CIFAR10-C experiments, see Figure 3. Highly luminous grids



(a) sub-model A, target (b) sub-model B, target (c) vanilla model, target

(d) sub-model A, shortcut (e) sub-model B, shortcut (f) vanilla model, shortcut

Figure 2: The t-SNE visualization of sub-models and a vanilla model as the control. Each point represents the feature extracted from a sample. The color of the point indicates the attributes of that sample. In the top row points are colored with target attribute label, and in the bottom row points are colored with shortcut attribute label. Note that shortcut labels are only used for visualization and are not available during training.

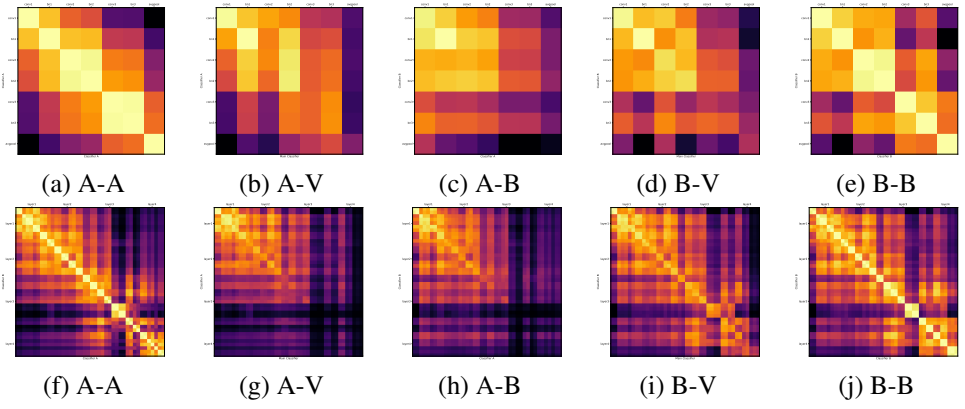
in the figure indicate that the two layers represented by the corresponding coordinates have a high degree of similarity while the dark color indicate low similarity.

We begin by observing the diagonal value of the results, as this indicates the similarity of the same layers of the two models and most directly reflects the similarity of the two models. For the case where the two models analyzed are the same model, the diagonal line indicates the similarity between each layer and itself, i.e., 1. It can be seen that sub-models A and B have the lowest similarity, such results suggest that the two sub-models learn very different features, which is consistent with our expect that the two sub-models model different attributes separately. The similarity between sub-model and the vanilla model is a little higher than that between sub-models, which indicates that the vanilla models learn some of the features modeled by both of the sub-models to some extent. In CMNIST, the results for the two sub-models A and B are almost symmetric. And in CIFAR10-C, sub-model B rather than sub-model A, is more similar to the vanilla model. A possible reason is that the CIFAR10-C features are more complex and difficult. It is difficult for the model to completely untangle the inter-attribute entanglement. In addition, we note that there is a higher similarity between different models at lower layers, which may be due to the fact that different attributes have shared low-level features.

The following conclusions can be drawn from the experiments concerning CKA.

- The proposed differentiation method can indeed find different sub-models from one model to achieve learning of different viattributes.

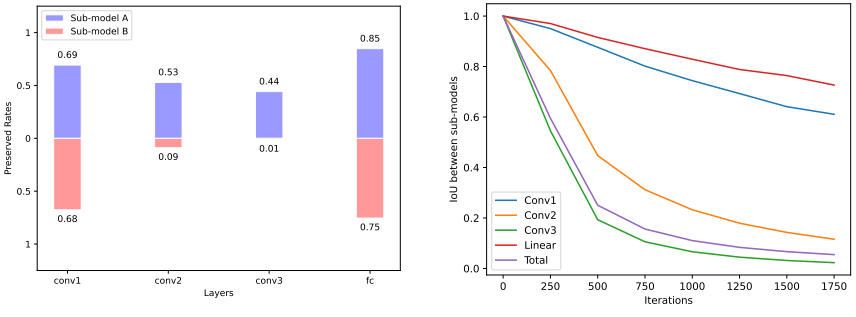
- Different sub-models also model some common low-level features, but the high-level



(a) A-A (b) A-V (c) A-B (d) B-V (e) B-B

(f) A-A (g) A-V (h) A-B (i) B-V (j) B-B

Figure 3: The visualzation of CKA value between models. Brighter grids indicate that the corresponding two layers have a higher degree of similarity. The top row shows the results on CMNIST, and the bottom row shows the results on CIFAR10-C. The ratio of both dataset is 0.5%. Which two models the results come from are specified in the subheadings, 'A', 'B', 'V' represents for sub-model A, sub-model B and vanilla model, respectively.



(a) Rate of reserved neurons (b) IoU between sub-models

Figure 4: Rate of reserved neurons and IoU between sub-models during training process.

features are largely independent.

We record the rate of reserved neurons and the IoU of the two masks during training as shown in Figure 4, the result is obtained on CMNIST(ratio=0.5%). It can be observed that the IoU of the layers from both sub-models are decreasing as the training progresses. For the conv layers, higher the feature layer has lower the IoU, which also corroborates with the results of CKA. Also the reserved rate is lower. We also notice that the reserved rate and the IoU are large for the linear layer, which we attribute to the fact that the linear layer itself has fewer parameters and lower redundancy. In feature layers with more parameters and higher sparsity, it will be easier for sub-models to differentiate.

5 Discussions

In this paper, we propose a strategy for constructing sub-model differentiation and implement it for training in biased datasets. We think that the strategy of learning multiple attributes separately is the way to go for solving out-of-distribution generalization tasks. There are two reasons behind this statement. Above all, shortcuts are not the same as bad attributes. In fact, shortcuts are somehow the "most cost-effective" features. In most task scenarios, the model prioritizes learning the simplest and most effective features. And for out-of-distribution generalization, whether these features are usable or not should depend on the actual situation. Thus, we suggest learn the different attributes separately and disentangle them. The assumption that simple features are not target features is too strong and can greatly limit the application. Another point is that specific task scenarios will provide the means for feature selection. Thus, the strategy is sufficient to solve most problems of out-of-distribution generalization. For example, in debiasing task, the predictive power of target attributes is larger than that of bias attributes. In summary, the proposed strategy is human logical and feasible.

6 Conclusion

In this paper, we propose a novel approach based on shortcut learning to address the challenges associated with data containing multiple attributes. Our approach involves identifying submodels within the main model, each of which is specialized in modeling a specific attribute. By constructing different sub-models using distinct masks, we achieve differentiated sub-model construction. This is accomplished through data redistribution during training and expanding the differences inherent in the masks themselves. Furthermore, we introduce a positive feedback mechanism that facilitates the continual expansion of differences among the sub-models. This mechanism ultimately enables each sub-model to learn separate attribute effectively. By learning the target and shortcut attributes independently, our proposed method demonstrates improved generalization performance on biased datasets. Moreover, its applicability can be extended to various datasets, positively impacting the model fairness and enhancing the generalization capabilities.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants U19B2044 and the Anhui Provincial Natural Science Foundation under Grant 2108085UD12. We acknowledge the support of GPU cluster built by MCC Lab of Information Science and Technology Institution, USTC.

References

[1] Sumyeong Ahn, Seongyoon Kim, and Se-Young Yun. Mitigating dataset bias by using per-sample gradient. In *NeurIPS ML Safety Workshop*, 2022.

[2] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning de-biased representations with biased representations. In Hal Daumé III and Aarti Singh, editors, *International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 528–539, 2020.

[3] Huanhuan Chen and Xin Yao. Regularized negative correlation learning for neural network ensembles. *IEEE Transactions on Neural Networks*, 20(12):1962–1979, 2009.

[4] Nikolay Dagaev, Brett D. Roads, Xiaoliang Luo, Daniel N. Barry, Kaustubh R. Patil, and Bradley C. Love. A too-good-to-be-true prior to reduce shortcut reliance. *Pattern Recognition Letters*, 166:164–171, 2023. ISSN 0167-8655. doi: https://doi.org/10.1016/j.patrec.2022.12.010.

[5] Wenhang Ge, Chunyan Pan, Ancong Wu, Hongwei Zheng, and Wei-Shi Zheng. Cross-camera feature prediction for intra-camera supervised person re-identification across distant scenes. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 3644–3653, 2021.

[6] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[7] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[8] Mahmoud Hassaballah and Ali Ismail Awad. *Deep learning in computer vision: principles and applications*. CRC Press, 2020.

[9] Zhen Huang, Xu Shen, Jun Xing, Tongliang Liu, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-Sheng Hua. Revisiting knowledge distillation: An inheritance and exploration framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3579–3588, 2021.

[10] Inwoo Hwang, Sangjun Lee, Yunhyeok Kwak, Seong Joon Oh, Damien Teney, Jin-Hwa Kim, and Byoung-Tak Zhang. Selecmix: Debiased learning by contradicting-pair sampling. *Advances in Neural Information Processing Systems*, 35:14345–14357, 2022.

[11] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, pages 1–12, 2016.

[12] Uday Kamath, John Liu, and James Whitaker. *Deep learning for NLP and speech recognition*, volume 84. Springer, 2019.

[13] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[14] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019.

[15] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

[17] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015. ISSN 1476-4687. doi: 10.1038/nature14539.

[18] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. Learning debiased representation via disentangled feature augmentation. In *Advances in Neural Information Processing Systems*, volume 34, pages 25123–25133. Curran Associates, Inc., 2021.

[19] Yingwei Li, Qihang Yu, Mingxing Tan, Jieru Mei, Peng Tang, Wei Shen, Alan Yuille, and cihang xie. Shape-texture debiased neural network training. In *International Conference on Learning Representations*, 2021.

[20] Ziqiang Li, Pengfei Xia, Rentuo Tao, Hongjing Niu, and Bin Li. A new perspective on stabilizing gans training: Direct adversarial training. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 7(1):178–189, 2022.

[21] Ziqiang Li, Muhammad Usman, Rentuo Tao, Pengfei Xia, Chaoyue Wang, Huanhuan Chen, and Bin Li. A systematic survey of regularization and normalization in gans. *ACM Computing Surveys*, 55(11):1–37, 2023.

[22] Jongin Lim, Youngdong Kim, Byungjai Kim, Chanho Ahn, Jinwoo Shin, Eunho Yang, and Seungju Han. Biasadv: Bias-adversarial augmentation for model debiasing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3832–3841, 2023.

[23] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 20673–20684, 2020.

[24] Hongjing Niu, Hanting Li, Feng Zhao, and Bin Li. Roadblocks for temporarily disabling shortcuts and learning new knowledge. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[25] Geon Yeong Park, Sangmin Lee, Sang Wan Lee, and Jong Chul Ye. Training debiased subnetworks with contrastive weight pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7929–7938, 2023.

[26] Mohammad Pezeshki, Oumar Kaba, Yoshua Bengio, Aaron C Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *Advances in Neural Information Processing Systems*, 34:1256–1272, 2021.

[27] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[28] Haohan Wang, Zexue He, and Eric P. Xing. Learning robust representations by projecting superficial statistics out. In *International Conference on Learning Representations*, 2019.

[29] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 834–843, 2021.

[30] Wei Zhang, Xianpeng Ji, Keyu Chen, Yu Ding, and Changjie Fan. Learning a facial expression embedding disentangled from identity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6759–6768, 2021.

[31] Yi-Kai Zhang, Qi-Wei Wang, De-Chuan Zhan, and Han-Jia Ye. Learning debiased representations via conditional attribute interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7599–7608, 2023.

[32] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, 31, 2018.