DEBUGGING CONCEPT BOTTLENECKS THROUGH INTERVENTION: SHORTCUT REMOVAL + RETRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

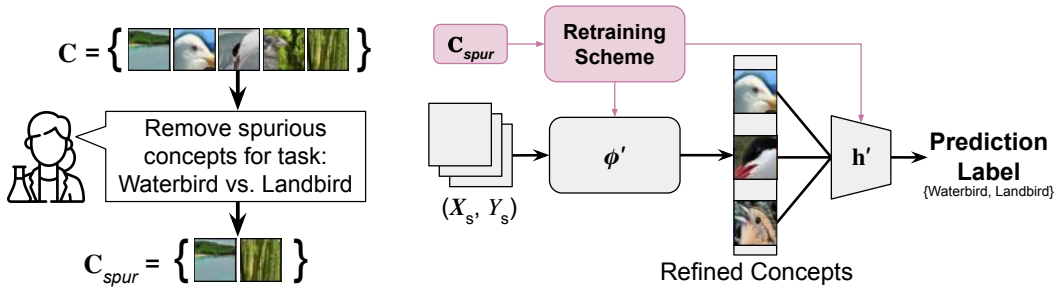
Machine learning models often learn unintended shortcuts (spurious correlations) that do not reflect the true causal structure of a task and thus degrade dramatically under subpopulation shift. This problem becomes especially severe in high-stakes domains where the cost of relying on misaligned shortcuts is prohibitive. To address this challenge, concept bottlenecks explicitly factor predictions into high-level concepts and a simple decision layer, enabling experts to diagnose whether learned concepts align with their domain knowledge. Yet, simply removing undesirable concepts after training is insufficient to prevent shortcuts when the concept encoder is incomplete or entangled. In this work, we propose *CBDebug*, a novel framework to debug concept bottlenecks for robustness under subpopulation shift. First, a domain expert identifies and removes spurious concepts using model explanations (the *Removal* step). Then, leveraging this human feedback, we disentangle or replace the removed shortcuts by retraining on a rebalanced dataset based on the causal graph (the *Retraining* step). Empirically, *CBDebug* significantly outperforms existing concept-based methods. Overall, our work demonstrates how expert-guided debugging of concept bottlenecks can achieve interpretability and robustness, promoting alignment of a model's internal reasoning with how humans reason.

1 INTRODUCTION

A critical roadblock in the deployment of machine learning systems in the real world is the fundamental misalignment between how humans reason and what machine learning models learn from data. Shortcut learning refers to the phenomenon where a model utilizes a spurious attribute which does not reflect the underlying causal relationship, such as using the presence of snow to label an image with a 'wolf' class (Geirhos et al., 2020). Such spurious attributes are often exposed when the proportion of subpopulations defined by these spurious attributes and the label changes in the test data, often referred to as subpopulation shift (Yang et al., 2023). We explore recent work on shortcut learning and subpopulation shift in Section A.2.1. In practice, these shifts are quite common, and models that do not map the true causal relationship between attributes fail drastically (Ye et al., 2024). In critical domains such as healthcare, we cannot reliably deploy a predictor that fails in unintuitive ways because there is a high cost of failure when individual lives are involved. We need machine learning models that capture a domain expert's intuition and are correct for the right reasons (Ross et al., 2017).

Training interpretable models help to bridge this reasoning gap. For vision classification, a popular interpretable framework is the concept bottleneck (Koh et al., 2020; Yuksekgonul et al.; Oikarinen et al.; Chen et al., 2019; Nauta et al., 2023b; Ma et al., 2024). A concept bottleneck first generates high-level concepts with an encoder ϕ and then passes these concepts through a simple layer h to predict the label, and we explore different architectures in more detail in Section A.2.2. Concept bottlenecks allow a user to investigate the set of learned concepts and evaluate the quality of the simple layer to see if the model's reasoning process is aligned with their intuition.

Moreover, concept bottlenecks enable the user to intervene at the concept-level to debug the model as we explore in more detail in Section A.2.3. The field of explanatory interactive machine learning (Teso & Kersting, 2019) has focused on this goal for general model explanations, and more recent works have focused on concept-level debugging (Bontempelli et al., 2021; 2023). However,



(a) Removal. User removes spurious concepts.

(b) Retraining. Update encoder ϕ and linear layer h based on user feedback.

Figure 1: Our debugging framework for incorporating a domain expert's knowledge into a concept bottleneck. **Removal (a)**: user marks each concept as spurious or core and corresponding weights in the linear layer are set to zero. Water and bamboo concepts are removed from the representation. **Retraining (b)**: Encoder and linear layer are retrained based on this human feedback to update concepts and linear mapping. Remaining core concepts are updated to remove any reliance on spurious concepts, such as entanglement between the bird's head and background.

these methods (Bontempelli et al., 2023) modify the loss to push the concept representation away from spurious concepts, which does not suffice to remove shortcuts, particularly when the learned representation is entangled or the model has overlooked necessary concepts in favor of shortcuts.

In this work, we investigate the problem of debugging concept bottlenecks to learn a predictor that is robust under subpopulation shift, through a general two-step process: **Removal** and **Retraining**.

Given a trained concept bottleneck, a user can evaluate the explanation for each concept and decide whether or not the concept is predictive of the label for the underlying classification task. For example, a radiologist may know that certain locations in an MRI scan are not predictive of a certain disease, and they could flag the corresponding concepts from the model. This enables a domain expert to inject their knowledge of the underlying classification task into the model by disallowing such spurious concepts. We call this the **Removal** step (Figure 1a). However, the removal of concepts does not suffice if the learned representation is not perfectly disentangled, or certain predictive concepts may have been overlooked due to the dominance of shortcuts in the learning process. To address this, we leverage user feedback to disentangle core concepts from the removed ones and construct a more comprehensive concept set. We refer to this process as **Retraining** (Figure 1b).

To effectively perform retraining, we propose CBDebug (Concept Bottleneck Debugger), a holistic approach that leverages causal reasoning to perform augmentation and permutation weighting to break unwanted correlations. We first utilize the feedback on undesired concepts to generate auxiliary variable labels for each instance based on the removed concept activations. Then, we perform shortcut removal through an augmentation and permutation weighting scheme to balance the training dataset to approximate the unconfounded distribution. By retraining on this balanced dataset, we can enhance the robustness of the concept bottleneck against the identified spurious concepts.

In summary, we highlight the benefits of training and debugging concept bottlenecks for robustness to subpopulation shift, proposing CBDebug, a causally-motivated approach that incorporates human feedback to remove spurious concepts from the model and retrains the model to further disentangle the learned concepts. We also CBDebug on a state-of-the-art interpretable model PIP-Net on Waterbirds, **improving the originally trained model's worst-group accuracy by 22.7%**.

2 Methodology

Problem Setting. Given a dataset $(\mathbf{X}_s, Y_s) \sim P_s$, a concept bottleneck trained on that dataset $\{\phi, h\}$, and a user with the knowledge of the classification task. We would like to utilize the domain expert to debug the model and return an updated concept bottleneck $\{\phi', h'\}$ that removes dependence on spurious concepts $\mathbf{V}$ from $\phi(\mathbf{X}_s)$ and utilizes non-spurious concepts in its predictions. We first detail our general framework for debugging a concept bottleneck as shown in Figure 1,

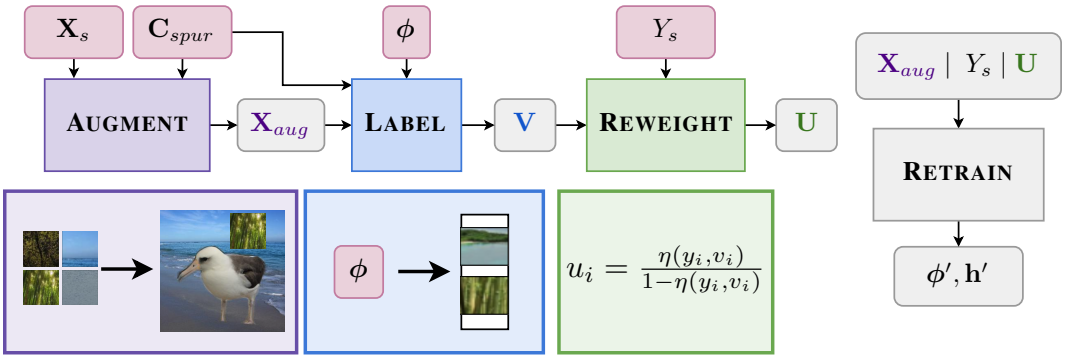


Figure 2: Overview of CBDebug (Concept Bottleneck Debugger), which consists of three main steps. First, augmentation is performed on $\mathbf{X}_s$ based on the spurious concepts $\mathbf{C}_{spur}$ to generate X_{aug}. Then, the encoder ϕ computes the concept activations for spurious concepts in $\mathbf{C}_{spur}$ to generate the auxiliary variable $\mathbf{V}$. Finally, reweighting is performed based on the auxiliary variable $\mathbf{V}$ and main label Y_s to compute the odds of the sample being drawn from the unconfounded distribution to generate weights $\mathbf{U}$. This new dataset $(\mathbf{X}_{aug}, Y_s)$ and weights $\mathbf{U}$ are used to retrain the concept bottleneck $\{\phi, h\} \rightarrow \{\phi', h'\}$.

then describe our approach CBDebug for effectively performing this retraining. Motivation for our approach can be found in A.1.

2.1 Our Debugging Framework

Our framework for efficiently debugging a concept bottleneck with a domain expert consists of two main steps: **Removal** and **Retraining**. We generalize ProtoPDebug's (Bontempelli et al., 2023) debugging framework to any concept bottleneck that is able to explain concepts only in the penultimate layer of the network.

Removal. This step aims to take the trained concept bottleneck $\{\phi, h\}$ and return a set of spurious concepts $\mathbf{C}_{spur}$ based on the user's feedback. To do this, we show the set of concepts $\mathbf{C}$ and their explanations to a domain expert, who selects a subset which are considered spurious, denoted by $\mathbf{C}_{spur}$ (Figure 1a). **Retraining.** In the retraining step, the goal is to take the training samples and labels $(\mathbf{X}_s, Y_s)$, the concept bottleneck $\{\phi, h\}$ trained on this dataset, and the set of spurious concepts $\mathbf{C}_{spur}$ marked in the removal step and return an updated concept bottleneck $\{\phi', h'\}$ which has removed the spurious concepts and is trained to instead use other concepts in its predictions. Our algorithm performs adapted fine-tuning on the original concept bottleneck and returns the updated concept bottleneck $\{\phi', h'\}$ as depicted in Figure 1b.

By retraining the concept bottleneck, we aim to include the domain expert's feedback into the encoder ϕ to learn a new concept set that is independent of the spurious concepts, and then train the linear layer h based on these new concepts.

2.2 CBDebug

In this section, we introduce our approach CBDebug (Concept Bottleneck Debugger) for effectively retraining the concept bottleneck based on user feedback. By interpreting the user feedback as information about the causal graph from Figure 4, we can directly perform shortcut removal on the undesired concepts. Furthermore, by passing the training dataset through the concept bottleneck and collecting the activations of spurious concepts, we can get a real-valued multi-dimensional label representing auxiliary factors of variation that the user does not believe are causal to the underlying classification task and would like the model to be invariant to. In addition, any concept marked as spurious will be correlated with the label Y in the training dataset, because the model would not learn to use it otherwise, directly representing the shortcuts we aim to remove.

Our approach, as shown in Figure 2, is composed of three major steps: augmentation, labeling, and reweighting. We first describe our augmentation approach that balances the initial dataset by

spreading spurious concepts across classes in Section 2.2.1, then describe our labeling approach to uncover the auxiliary factors of variation $\mathbf{V}$ in Section 2.2.2, then describe our reweighting approach to recover the unconfounded distribution in Section 2.2.3.

2.2.1 AUGMENTATION

In the augmentation step, we take the training samples $\mathbf{X}_s$ and perform an augmentation step based on the spurious concept set $\mathbf{C}_{spur}$ to return new training samples $\mathbf{X}_{aug}$ that reduce the correlation between spurious concepts and the label as shown in Figure 2. We assign each class all concepts in the concept bottleneck that have a nonzero connection to that class in the linear layer h. Then, for each sample in the training dataset we randomly select a concept from all classes that the sample does not belong to (weighted by the concept's connection strength in the linear layer) and augment the sample with that spurious concept. Importantly, because these concepts were explicitly marked as spurious by the user, we assume that the augmentation does not change the label.

For text-based concept bottlenecks, a concept bank is required such as utilized in DISC (Wu et al., 2023) and mixup (Zhang, 2017) can be performed with images from the concept bank to reduce the model's reliance on spurious concepts. However, for prototype-based models, we can directly use the prototypical patches representing each concept. We perform CutMix (Yun et al., 2019) by selecting k random concepts and for each concept randomly selecting one of the top ten patches that activate highest on that concept. We found empirically that first performing augmentation was effective in reducing the variance of the reweighting process by incorporating spurious patches into more samples and increasing the number of samples belonging to minority subgroups. This helps for severe attribute imbalance, where just performing reweighting alone may result in very high variance. However, we found that augmentation alone was unable to fully remove dependence on the shortcut, so we combine it with a more theoretically grounded approach for shortcut removal.

After augmenting the training dataset, we can utilize our main labeling and reweighting scheme to recover the idealized distribution based on the provided user feedback.

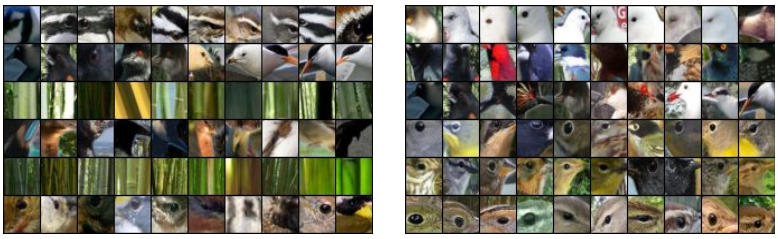
2.2.2 LABELING

In the labeling step, we take the augmented samples $\mathbf{X}_{aug}$ and the encoder ϕ and based on the spurious concept set $\mathbf{C}_{spur}$ we return a multi-dimensional auxiliary label V for each sample as shown in Figure 2. If we had a label for every sample in our dataset for all the auxiliary factors of variation V that were correlated with the label in our training dataset but not causally related according to Figure 4, we could apply (Zheng & Makar, 2022) which theoretically and empirically learns an optimal risk invariant predictor.

However, this approach has two main drawbacks. The first is the volume of human annotation required for labeling the auxiliary factors of variation. The second is the requirement that all auxiliary factors of variation must be known a priori. We tackle both of these limitations in this work through the use of a concept bottleneck. To label the auxiliary factors, we take the user feedback provided at the conceptual level and take the subset of concepts that were marked as spurious. Then, we take the spurious concept activations for each sample as the ground truth label for all auxiliary factors of variation. For example, if a user marks a 'bamboo' concept as spurious, then any sample that activates highly on that concept will likely have bamboo in its background. This reduces the annotation requirement by orders of magnitude from having to label each sample with multiple auxiliary factors of variation V to just having to label each concept in the concept bottleneck.

Additionally, even if auxiliary factors could be labeled automatically for specific datasets, by allowing a human in the loop we can better incorporate a domain experts knowledge into the model, which is critical for domains such as science and healthcare. The second main benefit of using prototype-based models that learn concepts directly from the data is that the user does not need to know a priori which factors of variation are undesired for the classification task. By simply investigating the explanation for each concept, the user can uncover any undesired biases that the model has learned from the training dataset and then remove those concepts.

Once we have generated auxiliary labels for each sample, we can perform data balancing by performing sample reweighting to remove the shortcut, and we describe this approach in the next section.



(a) Original (b) CBDebug

Figure 3: The six most highly activated concepts for the Original model trained on the dataset and the model after retraining with CBDebug. CBDebug removes both concepts representing bamboo from the concept set and replaces them with more robust concepts representing bird features.

2.2.3 REWEIGHTING

In the reweighting step, we take the auxiliary variables V and the labels Y and return importance weights U for each sample as shown in Figure 2.

We perform permutation weighting (Zheng & Makar, 2022) to recover the unconfounded distribution based on the auxiliary factors of variation that we labeled with our concept bottleneck in the previous step. The goal of permutation weighting is to reweight each sample based on how likely that the sample could have been drawn from the unconfounded distribution P° compared to P_s

To compute this likelihood, we collect the multi-dimensional auxiliary label V for each sample, as well as the label Y for the classification task. This dataset D represents the confounded distribution, where there exists a spurious correlation between the label Y and auxiliary label V. We create a new dataset D' by randomly permuting the label Y in the original dataset, which represents the unconfounded distribution where there is no correlation between Y and V.

We take the combination of V and Y as the features and label all samples in D with the label 0 (confounded distribution) and all samples in D' with the label 1 (unconfounded distribution) and train a binary predictor $\eta : Y \times \mathbf{V} \rightarrow \{0, 1\}$ on this new dataset. We then evaluate a weight u_i for each sample x_i by collecting the concept activations $v_i = \phi(x_i)$ (selecting only the concepts that belong to the spurious concept set $\mathbf{C}_{spur}$) and label y_i and then compute the odds that the sample belongs to the unconfounded distribution.

$$u_i = \frac{\eta(y_i, v_i)}{1 - \eta(y_i, v_i)} \tag{1}$$

To avoid having to hold out training data to perform this procedure, we perform K-fold cross validation and store the estimated weights only for the validation set. We also compute the average weight for each sample across multiple different permuted datasets as noted in (Arbour et al., 2021). We then reweight the classification loss for each sample to recover the idealized distribution as shown in (Makar et al., 2022).

3 EXPERIMENTS

To validate our approach, we evaluate CBDebug's ability to debug spurious correlations by testing on Waterbirds (Sagawa et al., 2019), showing our approach can more effectively remove shortcuts detected by a user than the previous state-of-the-art debugger (Bontempelli et al., 2023).

Case Study: Waterbirds. In Waterbirds, the goal is to predict whether the bird is a landbird or waterbird, but the labels are spuriously correlated with the background. For example, landbirds are more commonly seen on land and waterbirds more commonly seen on water in the training dataset, but this correlation is broken in the test dataset. As explored in (Yang et al., 2023), the Waterbirds dataset experiences multiple subpopulation shifts: spurious correlation, attribute imbalance, and class imbalance. We investigate the capabilities of different retraining methods to reduce the concept bottleneck's reliance on the background to improve its robustness to these subpopulation shifts.

Table 1: Average and worst-group accuracy across three runs for different retraining methods on the Waterbirds dataset. Best in **bold**, second best underlined.

	Average Acc	Worst-Group Acc
Original	87.2	59.3
Removal	88.4	72.0
Retraining	91.4 (0.4)	71.1 (2.0)
ProtoPDebug	90.9 (0.5)	71.8 (0.7)
Augment	91.2 (0.8)	74.8 (2.4)
Label + Reweight	**92.4 (0.4)**	73.6 (2.2)
CBDebug	91.6 (0.6)	**82.0 (1.8)**

To evaluate CBDebug, we utilize PIP-Net (Nauta et al., 2023b), a state of the art interpretable vision classification model that utilizes a self-supervised objective to first learn concepts from data, then learns to classify the classes from those concepts. We train PIP-Net with a ConvNeXt backbone on Waterbirds and it learns 134 concepts to make predictions. We show the six most activated concepts for PIP-Net in Figure 3, which shows two bamboo concepts that dominate the predictions of the model. We also evaluate its performance in Table 1. PIP-Net performs poorly (59.3% worst-group accuracy) on the minority subgroup in the training dataset: waterbirds on land, reiterating that interpretable models are still vulnerable to learning shortcuts in the data.

Then, we perform removal by marking all of the concepts that focus on the background into the spurious set $\mathbf{C}_{spur}$. We remove 31 spurious concepts out of the 134 original concepts. Surprisingly, this provides a 12.7% boost to worst-group accuracy, highlighting how well disentangled the learned concepts are and showing that the model still learned core features in addition to spurious features.

However, there is still a large gap between the average and worst-group accuracy, showing that the background shortcut used by the model has not been completely removed. We perform retraining based on this user feedback to improve the worst-group accuracy further. We evaluate the effectiveness of different retraining approaches as well as the individual components of our approach compared to CBDebug. All approaches first remove the spurious concepts, and then retrain the model for five epochs.

Retraining: Performing retraining after removing spurious concepts. **ProtoPDebug** (Bontempelli et al., 2023): Collect images in input space representing spurious concepts and store into a forget set. Add a regularizer to ensure the concepts in the forget set do not activate highly. **Augment**: Perform retraining on a dataset augmented with spurious concepts (Section 2.2.1). **Label + Reweight**: Perform permutation weighting to reweight dataset (Section 2.2.2) without performing augmentation.

We evaluate the performance of each retraining algorithm in Table 1. CBDebug is able to greatly outperform other retraining approaches on improving the worst-group accuracy of the model, providing a 10.0% boost in worst-group accuracy compared to concept removal. While Label + Reweight has higher average accuracy, CBDebug greatly improves the worst-group accuracy compared to either of its components by themselves, showing the benefits of combining the two data balancing methods. We also show in Figure 3 the new six most activated concepts for PIP-Net after retraining with CBDebug, showing the two bamboo concepts originally learned were removed and replaced with more robust concepts focusing on core attributes of the birds.

4 Conclusions

In this work, we looked at the connections between training interpretable models and model robustness, specifically under subpopulation shift. We present a causally-motivated approach for debugging concept bottlenecks to increase robustness to attribute imbalance and spurious correlations. When utilizing machine learning in critical domains, interpretability helps domain experts ensure the model is right for the right reasons by highlighting the reasoning process, but we also show in this work that interpretable models have multiple benefits for improving robustness to real-world distribution shifts.

REFERENCES

Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*, 2020.

Ibrahim Alabdulmohsin, Nicole Chiou, Alexander D'Amour, Arthur Gretton, Sanmi Koyejo, Matt J Kusner, Stephen R Pfohl, Olawale Salaudeen, Jessica Schrouff, and Katherine Tsai. Adapting to latent subgroup shifts via concepts and proxies. In *International Conference on Artificial Intelligence and Statistics*, pp. 9637–9661. PMLR, 2023.

David Arbour, Drew Dimmery, and Arjun Sondhi. Permutation weighting. In *International Conference on Machine Learning*, pp. 331–341. PMLR, 2021.

Alina Jade Barnett, Fides Regina Schwartz, Chaofan Tao, Chaofan Chen, Yinhao Ren, Joseph Y Lo, and Cynthia Rudin. A case-based interpretable deep learning model for classification of mass lesions in digital mammography. *Nature Machine Intelligence*, 3(12):1061–1070, 2021.

Usha Bhalla, Alex Oesterling, Suraj Srinivas, Flavio P Calmon, and Himabindu Lakkaraju. Interpreting clip with sparse linear concept embeddings (splice). *arXiv preprint arXiv:2402.10376*, 2024.

Andrea Bontempelli, Fausto Giunchiglia, Andrea Passerini, and Stefano Teso. Toward a unified framework for debugging concept-based models. *arXiv preprint arXiv:2109.11160*, 2021.

Andrea Bontempelli, Stefano Teso, Katya Tentori, Fausto Giunchiglia, Andrea Passerini, et al. Concept-level debugging of part-prototype networks. In *Proceedings of the The Eleventh International Conference on Learning Representations (ICLR 23)*. ICLR 2023, 2023.

Giacomo Capitani, Federico Bolelli, Angelo Porrello, Simone Calderara, and Elisa Ficarra. Clusterfix: A cluster-based debiasing approach without protected-group supervision. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4870–4879, 2024.

Zachariah Carmichael, Suhas Lohit, Anoop Cherian, Michael J Jones, and Walter J Scheirer. Pixel-grounded prototypical part networks. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4768–4779, 2024.

Rwiddhi Chakraborty, Adrian Sletten, and Michael C Kampffmeyer. Exmap: Leveraging explainability heatmaps for unsupervised group robustness to spurious correlations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12017–12026, 2024.

Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.

Jihye Choi, Jayaram Raghuram, Yixuan Li, Suman Banerjee, and Somesh Jha. Adaptive concept bottleneck for foundation models. In *ICML 2024 Workshop on Foundation Models in the Wild*.

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.

Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. *arXiv preprint arXiv:2204.02937*, 2022.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020.

Aaron Jiaxun Li, Robin Netzorg, Zhihan Cheng, Zhuoqin Zhang, and Bin Yu. Improving prototypical visual explanations with reward reweighing, reselection, and retraining. In *Forty-first International Conference on Machine Learning*, 2024.

Chiyu Ma, Jon Donnelly, Wenjun Liu, Soroush Vosoughi, Cynthia Rudin, and Chaofan Chen. Interpretable image classification with adaptive prototype-based vision transformers. *arXiv preprint arXiv:2410.20722*, 2024.

Maggie Makar, Ben Packer, Dan Moldovan, Davis Blalock, Yoni Halpern, and Alexander D'Amour. Causally motivated shortcut removal using auxiliary labels. In *International Conference on Artificial Intelligence and Statistics*, pp. 739–766. PMLR, 2022.

Mazda Moayeri, Wenxiao Wang, Sahil Singla, and Soheil Feizi. Spuriosity rankings: sorting data to measure and mitigate biases. *Advances in Neural Information Processing Systems*, 36:41572–41600, 2023.

Meike Nauta, Ron Van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14933–14943, 2021.

Meike Nauta, Johannes H Hegeman, Jeroen Geerdink, Jörg Schlötterer, Maurice van Keulen, and Christin Seifert. Interpreting and correcting medical image classification with pip-net. In *European Conference on Artificial Intelligence*, pp. 198–215. Springer, 2023a.

Meike Nauta, Jörg Schlötterer, Maurice Van Keulen, and Christin Seifert. Pip-net: Patch-based intuitive prototypes for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2744–2753, 2023b.

Fahimeh Hosseini Noohdani, Parsa Hosseini, Aryan Yazdan Parast, Hamidreza Yaghoubi Araghi, and Mahdieh Soleymani Baghshah. Decompose-and-compose: A compositional approach to mitigating spurious correlation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27662–27671, 2024.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. In *The Eleventh International Conference on Learning Representations*.

Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Jessica Schrouff, Alexis Bellot, Amal Rannen-Triki, Alan Malek, Isabela Albuquerque, Arthur Gretton, Alexander D'Amour, and Silvia Chiappa. Mind the graph when balancing data for fairness or robustness. *arXiv preprint arXiv:2406.17433*, 2024.

Seonguk Seo, Joon-Young Lee, and Bohyung Han. Unsupervised learning of debiased representations with pseudo-attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16742–16751, 2022.

Stefano Teso and Kristian Kersting. Explanatory interactive machine learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 239–245, 2019.

Katherine Tsai, Stephen R Pfohl, Olawale Salaudeen, Nicole Chiou, Matt Kusner, Alexander D'Amour, Sanmi Koyejo, and Arthur Gretton. Proxy methods for domain adaptation. In *International Conference on Artificial Intelligence and Statistics*, pp. 3961–3969. PMLR, 2024.

Shirley Wu, Mert Yuksekgonul, Linjun Zhang, and James Zou. Discover and cure: Concept-aware mitigation of spurious correlation. In *International Conference on Machine Learning*, pp. 37765–37786. PMLR, 2023.

Yuzhe Yang, Haoran Zhang, Dina Katabi, and Marzyeh Ghassemi. Change is hard: A closer look at subpopulation shift. *arXiv preprint arXiv:2302.12254*, 2023.

Wenqian Ye, Guangtao Zheng, Xu Cao, Yunsheng Ma, and Aidong Zhang. Spurious correlations in machine learning: A survey. *arXiv preprint arXiv:2402.12715*, 2024.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *The Eleventh International Conference on Learning Representations*.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Hongyi Zhang. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Jiayun Zheng and Maggie Makar. Causally motivated multi-shortcut identification and removal. *Advances in Neural Information Processing Systems*, 35:12800–12812, 2022.

A Appendix

A.1 Motivation

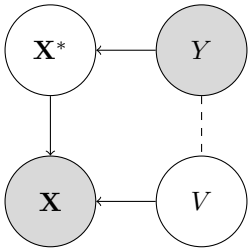


Figure 4: Causal DAG of the problem setting, adapted from Makar et al. (2022). The main label Y and auxiliary label $\mathbf{V}$ generate observed input $\mathbf{X}$, but Y's effect on $\mathbf{X}$ is only through $\mathbf{X}^*$. However, we do not require that $\mathbf{V}$ is observed and allow it to be multi-dimensional.

We assume access to samples $\mathbf{X}$ and labels Y, and assume that there exists some sufficient statistic $\mathbf{X}^*$ that captures all core features for predicting Y. We assume that our problem setting can be encapsulated by the causal DAG in Figure 4, where $\mathbf{V}$ denote the set of spurious attributes which are not causally related but may be correlated in the source distribution P_s used to train the model. Our problem setting is similar to Makar et al. (2022), however we allow for $\mathbf{V}$ to be multi-dimensional and do not require it to be observed.

Based on the causal graph, the training data follows the following distribution.

$$\mathcal{P}_s(\mathbf{X}, \mathbf{X}^*, \mathbf{V}, Y) = P_s(\mathbf{X}|\mathbf{X}^*, \mathbf{V})P_s(\mathbf{X}^*|Y)P_s(Y)P_s(\mathbf{V}|Y)$$

We assume there are some family of distributions where only the correlation between Y and $\mathbf{V}$ changes to $P_t(\mathbf{V}|Y)$.

$$\mathcal{P}_t = \{P_s(\mathbf{X}|\mathbf{X}^*, \mathbf{V})P_s(\mathbf{X}^*|Y)P_s(Y)P_t(\mathbf{V}|Y)\}$$

We now define a decomposed classifier, which consists of an encoder ϕ and a linear layer h to predict the output label.

Definition A.1 (Decomposed Classifier)**.** Assume $f : X \to Y$ where $f(X) = h(\phi(X))$. $\phi(X)$ maps an input X to an intermediate representation and h is a linear layer that maps the intermediate representation to the output.

We define the optimal risk invariant predictor f^* under subpopulation shift as follows, where the correlation between Y and $\mathbf{V}$ can change since there is no causal relationship.

$$f^* = \arg\min_f \sup_{P_t(\mathbf{V}|Y)} \mathbb{E}_{(\mathbf{x},y)\ P_t}[\ell(y, f(\mathbf{x}))]$$

Our objective maps to the one presented in (Sagawa et al., 2019) assuming the classes are balanced (i.e. $P_s(Y) = P_t(Y)$), accounting for spurious correlations as well as attribute imbalance and attribute generalization.

A common approach to learn f^* is to take some classifier $\{\phi, h\}$ and retrain h based on a balanced validation set to linearly project out $\mathbf{V}$ (Kirichenko et al., 2022). For this retraining to work, h must be able to linearly project out all dependence on $\mathbf{V}$ in the intermediate representation $\phi(X)$. (Schrouff et al., 2024) refers to this as the disentanglement of $\phi(X)$. When this requirement is not satisfied (i.e. the feature representation is low quality), last layer retraining as performed in (Kirichenko et al., 2022) will fail. This is also seen in (Yang et al., 2023) where classifier learning cannot improve attribute imbalance or attribute generalization effectively, since the core features were not learned by ϕ.

In this work, we highlight the benefits of training interpretable models for improving the disentanglement of $\phi(X)$ and overall robustness of the classifier f. We assume that the decomposed

classifier follows a concept bottleneck structure, where $\phi(X)$ maps to a vector of concept scores, and h maps those concept scores to the predicted labels.

The core requirement for a concept bottleneck is that each concept has a corresponding explanation as can be seen in Figure 1a. For example, the explanation for classic concept bottleneck models (Koh et al., 2020) is the textual description of the feature used in training, whereas for ProtoPNet (Chen et al., 2019) the explanation is the image patch whose embedding is used to compute concept activation scores. However, this explanation requirement is general and we highlight how different model architectures generate such explanations in Section A.2.2.

Training a concept bottleneck has two main benefits for robustness.

1. The primary goal of training interpretable models is to align the model's reasoning process with human reasoning. To achieve this alignment, the concept set is refined towards X^* through techniques such as augmentation, self-supervised learning, and concept regularization to ensure high quality concepts.
2. Concept bottlenecks allow users to better understand how the model makes predictions and enable interventions on this reasoning process by explicitly labelling spurious concepts.

In this work, we focus on using the labels provided by the user to retrain the model and learn a better disentangled concept set.

A.2 Related Work

In this section, we contrast our work with prior techniques on shortcut learning, concept bottleneck and debugging ML models.

A.2.1 Shortcut Learning and Subpopulation Shift

Shortcut learning presents a large roadblock in the deployment of machine learning systems in the real world (Geirhos et al., 2020). Trained models frequently learn shortcuts that do not hold under subpopulation shift, which allows for the proportion of subpopulations to differ at test time. Many prior methods have studied this problem (Sagawa et al., 2019; Kirichenko et al., 2022; Seo et al., 2022; Chakraborty et al., 2024; Capitani et al., 2024), as discussed in the recent benchmark paper (Yang et al., 2023). Additionally, theoretical connections between robustness and causality have been explored (Schrouff et al., 2024; Tsai et al., 2024; Alabdulmohsin et al., 2023), and some methods utilize the causal graph to perform shortcut removal (Makar et al., 2022; Zheng & Makar, 2022).

In this work, we remove the requirement for auxiliary variables labeled a priori by utilizing an interpretable model that learns concepts directly from the data and allowing a domain expert to intervene on the learned concepts.

A.2.2 Concept Bottleneck

Many different approaches fall under our definition of a concept bottleneck. The first is classic concept bottleneck models (Koh et al., 2020) and further improvements such as label-free CBMs (Oikarinen et al.), post-hoc CBMs (Yuksekgonul et al.), and SpLiCE (Bhalla et al., 2024). These approaches perform global intervention on the concepts to highlight the capabilities of a user to change the model's behavior, but do not focus on debugging. These methods map to text-based concepts in the concept bottleneck, and then train a sparse linear layer to make predictions based on the concept activations.

Another class of models is prototype-based approaches such as ProtoPNet (Chen et al., 2019) and its derivatives such as PIP-Net (Nauta et al., 2023b), ProtoViT (Ma et al., 2024), ProtoTree (Nauta et al., 2021), and PIXPNET (Carmichael et al., 2024). These approaches utilize image patches as the concepts in the concept bottleneck, and automatically learn these images patches without additional supervision.

Additionally, even class activation maps (Zheng & Makar, 2022) factor into the concept bottleneck structure because each neuron in the penultimate layer has a corresponding explanation.

A.2.3 DEBUGGING

Many works have focused on allowing a human to debug a machine learning model. A lot of initial work focused on explanatory interactive machine learning (Teso & Kersting, 2019). Multiple approaches utilize class activation maps to reduce spuriosity (Moayeri et al., 2023), identify causal components (Noohdani et al., 2024), or identify specific bugs (Adebayo et al., 2020).

A very related approach is DISC (Wu et al., 2023), which aims to reduce spurious correlations in the model through an iterative discovery and cure process. They focus on debugging a general black-box model automatically, while we focus on injecting human knowledge into the model.

Focusing on methods that are interpretable by design, (Nauta et al., 2023a) evaluates PIP-Net on two medical datasets and show they can find and remove concepts that represent spurious correlations, while IAIA-BL (Barnett et al., 2021) queries domain experts for object segmentation maps to provide extra supervision. Adaptive concept bottlenecks also focuses on improving the robustness of the model (Choi et al.), but they focus on test-time adaptation instead of utilizing user feedback.

Bontempelli et al. (2021) presents general ideas on debugging concept bottlenecks, but does not evaluate any approach, while ProtoPDebug (Bontempelli et al., 2023) utilizes concept level supervision on ProtoPNet (Chen et al., 2019) and performs a regularized retraining. R3-ProtoPNet extends the user feedback to a more extensive reward model that better captures concept utility (Li et al., 2024). We present a novel causally-motivated debugging approach connecting the goal of debugging to robustness to subpopulation shift.