POST-HOC CONCEPT BOTTLENECK MODELS

Mert Yuksekgonul, Maggie Wang, James Zou
Stanford University
{merty,maggiewang,jamesz}@stanford.edu

ABSTRACT

Concept Bottleneck Models (CBMs) map the inputs onto a concept bottleneck and use the bottleneck to make a prediction. A concept bottleneck enhances interpretability since it can be investigated to understand what concepts the model ”sees” in an input and which of these concepts are deemed important. However, CBMs are restrictive in practice as they require concept labels during training to learn the bottleneck. Additionally, it is questionable if CBMs can match the accuracy of an unrestricted neural network trained on a given domain, potentially reducing the incentive to deploy them in practice. In this work, we address these two key limitations by introducing Post-hoc Concept Bottleneck models (P-CBMs). We show that we can turn any neural network into a P-CBM without sacrificing model performance while still retaining interpretability benefits. Finally, we show that editing P-CBMs without any fine-tuning or use of data from the target domain can provide significant performance gains.

1 INTRODUCTION

There is growing interest in developing deep learning models that are interpretable and yet still flexible. One approach is concept analysis (Kim et al., 2018), where the goal is to understand if and how high-level human-understandable features are “engineered” and used by neural networks (see Related Works in Appendix A for a broader overview). For instance, we can probe a skin lesion classifier to understand if the *Atypical Pigment Networks* concept is encoded in the embedding space and used later to make the prediction.

Concept bottlenecks are inspired by the idea that we can solve the task of interest by applying a simple interpretable function (e.g. a sparse linear model or a decision tree) to an underlying set of human-interpretable concepts. For instance, when trying to classify whether a skin tumor is malignant, dermatologists look for different visual patterns, e.g. existence of *Blue-Whitish Veils* are demonstrated to be the most useful indicator of melanoma (Menzies et al., 1996; Lucieri et al., 2020).

By constraining the model to only rely on a set of concepts and an interpretable predictor, we can

1. **Explain** what information the model is using when classifying an input by looking at the weights/rules in the interpretable predictor.
2. **Understand** when the model made a particular mistake due to incorrect concept predictions.
3. **Intervene** on the bottleneck to fix false concept predictions and thus fix the mistake.
4. **Edit** the interpretable predictor to improve performance and generalizability.

Our work builds on the earlier idea of concept bottlenecks, specifically Concept Bottleneck Models (CBMs) (Koh et al., 2020). CBMs train an entire model in an end-to-end fashion by first predicting concepts, then using these concepts to predict the label. While CBMs provide many of the benefits mentioned above, they require access to concept labels during model training, i.e. each input must be annotated with which concepts are present. Even though there are a number of densely annotated datasets such as CUB(Welinder et al., 2010), this is particularly restrictive for real-world use cases. In practice, datasets rarely have concept annotations. Most state-of-the-art models are trained using very large datasets (Deng et al., 2009; Radford et al., 2021) only annotated with the task label.

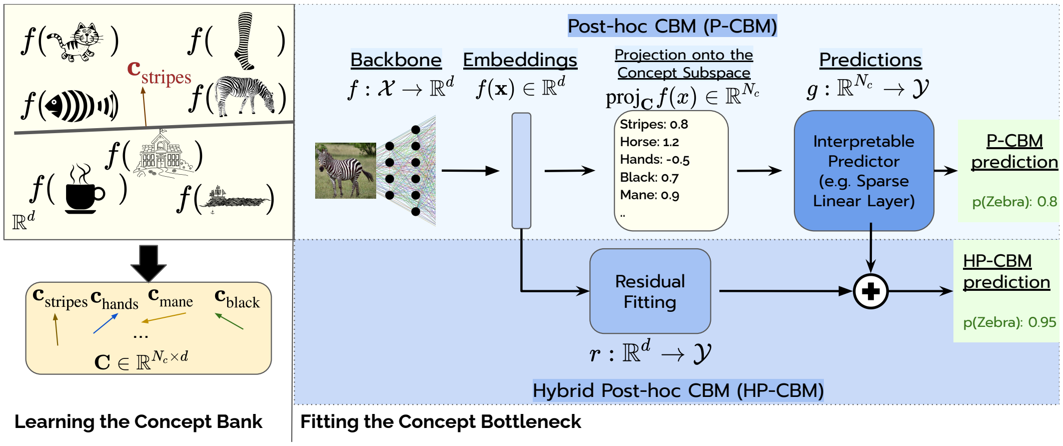

$f($ $)$
$\mathbf{c}_{\text{stripes}}$
$\mathbb{R}^d$
$\mathbf{c}_{\text{stripes}}$ $\mathbf{c}_{\text{hands}}$ $\mathbf{c}_{\text{mane}}$ $\mathbf{c}_{\text{black}}$
...
$\mathbf{C} \in \mathbb{R}^{N_c \times d}$
Learning the Concept Bank
Post-hoc CBM (P-CBM)
Backbone
$f : \mathcal{X} \to \mathbb{R}^d$
Embeddings
$f(\mathbf{x}) \in \mathbb{R}^d$
Projection onto the Concept Subspace
$\text{proj}_{\mathbf{C}} f(x) \in \mathbb{R}^{N_c}$
Predictions
$g : \mathbb{R}^{N_c} \to \mathcal{Y}$
Stripes: 0.8
Horse: 1.2
Hands: -0.5
Black: 0.7
Mane: 0.9
..
Interpretable Predictor (e.g. Sparse Linear Layer)
P-CBM prediction
p(Zebra): 0.8
Residual Fitting
$r : \mathbb{R}^d \to \mathcal{Y}$
HP-CBM prediction
p(Zebra): 0.95
Hybrid Post-hoc CBM (HP-CBM)
Fitting the Concept Bottleneck

Figure 1: **Post-hoc Concept Bottleneck Models. a)** First, we learn the CAVs corresponding to the concepts in our concept library. For each concept, e.g. stripes, we train a linear SVM to distinguish the embeddings of examples that contain the concept from those that do not. The CAV is defined as the vector normal to the classification boundary. Collectively, the set of CAVs define a concept subspace, and we project the embeddings produced by the pretrained backbone onto this new subspace. Second, we train an interpretable predictor to classify the examples based on their projections. When the concept library is incomplete, we can construct an HP-CBM by optionally introducing a residual predictor that maps the original embeddings to the target space.

Furthermore, for many downstream applications, the transfer learning approach, i.e. fine-tuning a pretrained model on the the downstream task, generally outperforms training from scratch. In addition, these fine-tuned models are more robust to distribution shifts (Hendrycks et al., 2020; Mathis et al., 2021; Hendrycks et al., 2019).

In this work, we propose the **Post-hoc Concept Bottleneck Model (P-CBM)** and **Hybrid P-CBM (HP-CBM)**. P-CBMs can convert any pre-trained model into a concept bottleneck model, and enhance the model with the desired interpretability benefits. HP-CBM is inspired by semiparametric models on fitting residuals (Härdle et al., 2004), and when the concept bottleneck is not rich enough, it adds a residual modeling step to recover the original model performance. In experiments across several tasks, we show that P-CBMs can be used without a loss in the original model performance. We further show that P-CBMs offer model edits without any fine-tuning or optimization.

2 Post-hoc Concept Bottleneck Models

There are two main steps while building a P-CBM, and an optional additional step to model the residuals that cannot be explained by the concept bottleneck. We let $f : \mathcal{X} \to \mathbb{R}^d$ be any pretrained backbone model, where d is the size of the corresponding embedding space and $\mathcal{X}$ is the input space. For instance, f can be the image encoder of CLIP (Radford et al., 2021) or the model layers up to the penultimate layer of a ResNet (He et al., 2016). An overview of the model can be found in 1, and we describe the steps in detail below.

Learning the Concept Subspace ($\boldsymbol{C} \in \mathbb{R}^{N_c \times d}$): To learn concept representations, we make use of CAVs (Concept Activation Vectors) (Kim et al., 2018). In particular, we first define a concept library $I = \{i_1, i_2, ..., i_n\}$. The concepts in the library can be selected by a domain expert, or learned automatically from the data (Ghorbani et al., 2019; Yeh et al., 2020). For each concept i, we collect embeddings for the positive examples, denoted by the set $P_i = \{f(x_{p_1}), ..., f(x_{p_{N_p}})\}$, that exhibit the concept, and negative examples $N_i = \{f(x_{n_1}), ..., f(x_{p_{N_n}})\}$ that do not contain the concept. Importantly, note that unlike CBMs, these samples can be different from the data used to train the backbone model. Following (Kim et al., 2018), we train a linear SVM using P_i and N_i to learn the corresponding CAV, that is, the vector normal to the linear classification boundary. We denote the CAV for concept i by $\boldsymbol{c}_i$. Let $\boldsymbol{C} \in \mathbb{R}^{N_c \times d}$ denote a matrix of concept vectors, where N_c is the number of concepts and each row $\boldsymbol{c}_i$ represents a concept. Given an input, we project the embedding of the input onto the subspace spanned by concept vectors (the concept subspace). Particularly, we let $f_{\boldsymbol{C}}(x) = \text{proj}_{\boldsymbol{C}} f(x)$, where the ith entry is $f_{\boldsymbol{C}}^{(i)}(x) = \frac{\langle f(x), \boldsymbol{c}_i \rangle}{||\boldsymbol{c}_i||_2^2} \boldsymbol{c}_i$.

Learning the Interpretable Predictor: Next, we define an interpretable predictor that maps the concept subspace to the model prediction. Concretely, let $g : \mathbb{R}^{N_c} \to \mathcal{Y}$ be an interpretable predictor, such as a sparse linear model or a decision tree, where $\mathcal{Y}$ denotes the label space. An interpretable predictor is desirable because it provides insight into which concepts the model is relying on when making a decision. If a domain expert observes a counter-intuitive phenomenon in the predictor, they can edit the predictor to improve the model. We demonstrate these benefits in our experiments.

To learn the P-CBM, we solve the following problem:

$$\min_{g} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(g(f_{\boldsymbol{C}}(x)), y)] + \lambda\Omega(g) \tag{1}$$

where $f_{\boldsymbol{C}} = \text{proj}_{\boldsymbol{C}} f(x)$, $\mathcal{L}(\hat{y}, y)$ is a loss function such as cross-entropy loss, $\Omega(g)$ is a complexity measure to regularize the model, and λ is the regularization strength. In this work, we use sparse linear models to learn the interpretable predictor, where $g(x) = \boldsymbol{w}^T x + b$. Similarly, we define $\Omega(g) = (\alpha||\boldsymbol{w}||_1 + (1-\alpha)||\boldsymbol{w}||_2^2)$ as the elastic-net penalty parameterized by α.

Residual Modeling: What happens when the concept bank is not sufficiently expressive? For instance, there may be skin lesion descriptors that are not available in the concept library. Ideally, we would like to preserve the original model accuracy while retaining the interpretability benefits. Drawing inspiration from the semiparametric models on fitting residuals (Härdle et al., 2004), we introduce **Hybrid Post-hoc CBMs(HP-CBM)**. Particularly, after fixing the concept bottleneck and the interpretable predictor, we re-introduce the embeddings to 'fit the residuals'. Particularly, we solve the following:

$$\min_{r} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(g(f_{\boldsymbol{C}}(x)) + r(f(x)), y)] \tag{2}$$

where $r : \mathbb{R}^d \to \mathcal{Y}$ is the residual predictor. We hypothesize that the residual predictor will compensate for what is missing from the concept bank, and recover the original model accuracy. We implement the residual predictor as a linear model, i.e. $r(x) = w_r^T x + b_r$. Note that in Equation 2, while training the residual predictor, the trained concept bottleneck is kept fixed. Given a trained HP-CBM, if we would like to observe model performance in the absence of the residual predictor, we can simply drop r.

3 Experiments

We evaluate P-CBMs and HP-CBMs in challenging image classification and medical settings, where we demonstrate intriguing reliability gains. First, we address practical concerns and show that P-CBMs can be used without a loss in the original model performance. Finally, we show the P-CBMs offer model edits without any fine-tuning or needing data from the target domain.

3.1 Concept bottlenecks do not hurt the full model performance

CIFAR10, CIFAR100 (Krizhevsky, 2009) We use CLIP-ResNet50(Radford et al., 2021) as the backbone model for CIFAR experiments, following linear probing methodology to evaluate the original model. For the concept bottleneck, we use 170 concepts introduced in (Abid et al., 2021) which are extracted from the BRODEN visual concepts dataset (Fong & Vedaldi, 2018). These concepts include objects (e.g. *dog*), settings (e.g. *snow*) textures (e.g. *stripes*), and (d) image qualities (e.g. *blurriness*). The full list of concepts can be found in (Abid et al., 2021).

CUB (Welinder et al., 2010) In the 200-way bird identification task, we use a ResNet18(He et al., 2016) trained on the CUB dataset, available in (osmr, 2022). We use training/validation splits shared in (Koh et al., 2020). We use the 312 concepts provided in the CUB dataset, which indicate various features including *wing shape*, *back pattern*, *eye color*, and more.

HAM10000 (Tschandl et al., 2018) is a dataset of dermoscopic images, which contain skin lesions from a representative set of diagnostic categories. The task we follow is detecting whether a skin lesion is benign or malignant. We use the Inception(Szegedy et al., 2015) model trained on this dataset, which is available from (Daneshjou et al., 2021). Following the setting in (Lucieri et al., 2020), we collect concepts from the Derm7pt (Kawahara et al., 2018) dataset. 8 concepts obtained from this dataset include *Blue Whitish Veil*, *Pigmented Networks*, *Regression Structures*, which are reportedly associated with the malignancy of a lesion. We further add the *dark-skin-color* concept to test for biases in these models, where we obtain the concept bank images from the Fitzpatrick17k(Groh et al., 2021) dataset.

SIIM-ISIC(Rotemberg et al., 2021) To test a real-world transfer learning use case, we evaluate the model trained on HAM10000 on a subset of the SIIM-ISIC Melanoma Classification dataset. We use 2000 images for training (400 malignant, 1600 benign) and evaluate the model on a held-out set of 500 images (100 malignant, 400 benign). We evaluate the original model performance using a linear probe. We use the same concepts described in the HAM10000 dataset.

For each of these datasets, we first train the P-CBM for 10 epochs. Next, we train the HP-CBM for 10 more epochs using the residual fitting step. Finally, we compare P-CBMs and HP-CBMs to the performance of the original model on the held-out test datasets. We use Adam(Kingma & Ba, 2015) to solve the optimization problems. All of the experiments use the same set of hyperparameters, where we have regularization strength $\lambda = 0.05$, $\alpha = 0.99$, and learning rate 0.01.

In Table 1, we report results over these five datasets. P-CBMs achieve comparable performance to the original model in all datasets except CIFAR100, and HP-CBMs match the performance in all scenarios. In CIFAR100, we hypothesize that the concept bank available is not sufficient to classify finer-grained classes, and hence there is a performance gap between the P-CBM and the original model. When the concept bank is not sufficient to solve the task, HP-CBMs can be introduced to recover the original model performance while retaining the benefits of P-CBMs (see next sections).

	CIFAR10	**CIFAR100**	**CUB**	**HAM10000**	**ISIC**
Original Model	0.888	0.701	0.612	0.891	0.830
P-CBM	0.705	0.432	0.596	0.887	0.828
HP-CBM	0.879	0.684	0.635	0.912	0.832
Change in Accuracy	-0.009	-0.017	0.023	0.021	0.002

Table 1: **Post-hoc Concept Bottlenecks do not hurt model performance**. Over five evaluations, P-CBMs/HP-CBMs do not exhibit a significant performance degradation. In CIFAR100, P-CBM performs poorly since the concept bank is not expressive enough to solve a finer-grained task; however, HP-CBMs recover original model accuracy. Strikingly, both P-CBM and HP-CBMs show improvements over the original model in HAM10000 and ISIC. Original CBMs cannot be trained on CIFAR/HAM10000/ISIC, as they do not have concept labels in the training dataset.

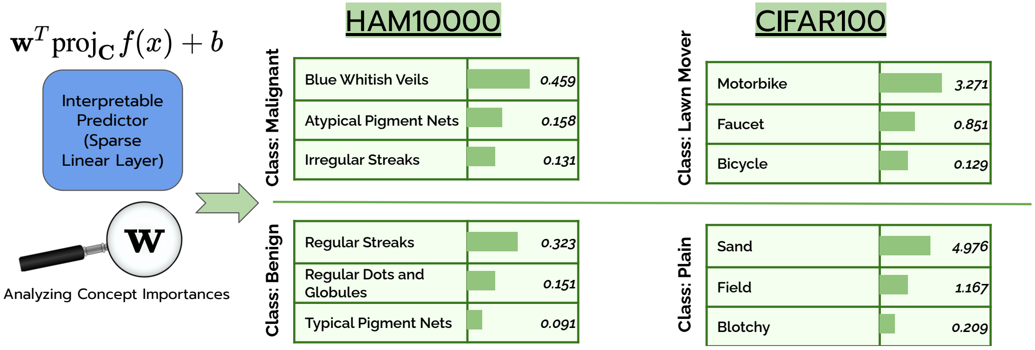


Figure 2: **Explaining Post-hoc CBMs. a)** We report the top 3 largest weights in the linear layer for the shown classes. For instance, *Blue Whitish Veils*, *Atypical Pigment Networks* and *Irregular Streaks* have the large weights for classifying whether a skin lesion is malignant. These are consistent with dermatologists' findings (Menzies et al., 1996).

In Figure 2, we provide sample concept weights in the corresponding P-CBMs. For instance in HAM10000, P-CBMs use *Blue Whitish Veils*, *Atypical Pigment Networks* and *Irregular Streaks* to identify malignant lesions, whereas *Typical Pigment Networks*, *Regular Streaks* and *Regular Dots* *and Globules* are used to identify benign lesions. These associations are in parallel with medical knowledge(Menzies et al., 1996; Lucieri et al., 2020). We observe similar cases in CIFAR-10 and CIFAR-100, where the class keyboard is associated with the concepts *keyboard*, *computer* and *mouse*; while the class lamp is associated with the concepts *lamp*, *light* and *candlesticks*.

We showed that P-CBMs and HP-CBMs are drop-in replacements for existing models with almost no performance loss. However, P-CBMs provide further benefits, as we demonstrate in the next section.

3.2 Model editing with Post-hoc Concept Bottlenecks

When we observe a spurious correlation in the concept bottleneck, can we make the model perform better by simple edit operations? In this section, we show that P-CBMs come with the benefit of 'free' model edits. Firstly, unlike many existing model editing approaches (see Appendix A), we do not assume any knowledge about the test domain. *We also do not need any data from either the train or target domains to perform the model edit*, which can be a significant advantage in practice when data is inaccessible. Given a trained P-CBM, we edit our concept bottleneck by just looking at the concept weights. For our editing experiments, we use the Metashift(Liang & Zou, 2022) dataset to simulate distribution shifts. We use 10 different scenarios where there is a distribution shift for a particular class between the training and test datasets. For instance, during training, we use table images where there is also a dog in the image, and test the model with table images where there is not a dog but a cat in the image. We denote the training domain as *table(dog)*, and the test domain as *table(cat)*. We give more details and the results of all domains in the Appendix C.

Given a P-CBM, we evaluate three editing strategies:

1. **Prune**: We set the weight of a particular concept on a particular class prediction to 0, i.e. for a class indexed by k and concept indexed by i, we let $\boldsymbol{w}_i'^{(k)} = 0$.

2. **Prune+Normalize**: After applying pruning, we rescale the concept weights. Let $\boldsymbol{w}'^{(k)}$ denote the pruned weights for class k. We renormalize the weights to match the original norms, such that $|\boldsymbol{w}'^{(k)}| = |\boldsymbol{w}^{(k)}|$. The normalization step alleviates the imbalance between class weights upon pruning a concept with large weights for a particular class.

3. **Fine-tune (Oracle)**: We compare our simple editing strategies to retraining, which can be considered an oracle. Particularly, we fine-tune our P-CBM using samples from the test domain, and then test the fine-tuned model with a set of held-out samples.

In the context of Metashift experiments, we simply edit the concept spuriously correlated with a particular class. Namely, for the domain *table(dog)*, we prune the weight of the *dog* concept for the class *table*. In Table 2, we report the result of our editing experiments over 10 scenarios. We observe that for P-CBMs, the Prune+Normalize strategy can recover 50% of the accuracy gains given by fine-tuning on the original test domain.

	Unedited	Prune	Prune + Normalize	Fine-tune(Oracle)
P-CBM Accuracy	0.656 ± 0.079	0.686 ± 0.081	0.750 ± 0.059	0.859 ± 0.09
P-CBM Edit Gain	-	0.029 ± 0.052	0.093 ± 0.078	0.202 ± 0.067
HP-CBM Accuracy	0.657 ± 0.122	0.672 ± 0.103	0.713 ± 0.086	0.861 ± 0.102
HP-CBM Edit Gain	-	0.017 ± 0.028	0.058 ± 0.055	0.190 ± 0.153

Table 2: **Model edits with Post-Hoc CBMs.** We report results over 10 distribution shift experiments generated using Metashift. We observe that very simple editing strategies in the concept subspace provide 50% of the gains made by fine-tuning on the test domain.

Even though our edit strategy is extremely simple, we can recover 50% of the gains made by fine-tuning the model. It is particularly easy-to-use since it can be applied without fine-tuning or using any knowledge or data from the target domain.

4 Limitations and Conclusion

In this work, we presented Post-hoc CBMs as a way of converting any model into a CBM, retaining the original model performance without losing the interpretability benefits. Many benefits of CBMs depend heavily on the quality of the concept library. Finding concept subspaces in an unsupervised fashion is an active area of research that will help with usability of concept bottlenecks. Additionally, future work will focus on better model editing approaches with concept subspaces.

References

Abubakar Abid, Mert Yuksekgonul, and James Zou. Meaningfully explaining model mistakes using conceptual counterfactuals. *arXiv preprint arXiv:2106.12723*, 2021.

Arjun Akula, Shuai Wang, and Song-Chun Zhu. Cocox: Generating conceptual and counterfactual explanations via fault-lines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

David Bau, Steven Liu, Tongzhou Wang, Jun-Yan Zhu, and Antonio Torralba. Rewriting a deep generative model. In *European Conference on Computer Vision*, pp. 351–369. Springer, 2020.

Roxana Daneshjou, Kailas Vodrahalli, Weixin Liang, Roberto A Novoa, Melissa Jenkins, Veronica Rotemberg, Justin Ko, Susan M Swetter, Elizabeth E Bailey, Olivier Gevaert, et al. Disparities in dermatology ai: Assessments using diverse clinical images. *arXiv preprint arXiv:2111.08006*, 2021.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Ruth Fong and Andrea Vedaldi. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8730–8738, 2018.

Asma Ghandeharioun, Been Kim, Chun-Liang Li, Brendan Jou, Brian Eoff, and Rosalind W Picard. Dissect: Disentangled simultaneous explanations via concept traversals. *arXiv preprint arXiv:2105.15164*, 2021.

Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in Neural Information Processing Systems*, 32, 2019.

Matthew Groh, Caleb Harris, Luis Soenksen, Felix Lau, Rachel Han, Aerin Kim, Arash Koochek, and Omar Badri. Evaluating deep neural networks trained on clinical images in dermatology with the fitzpatrick 17k dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1820–1828, 2021.

Wolfgang Härdle, Marlene Müller, Stefan Sperlich, and Axel Werwatz. *Nonparametric and semiparametric models*, volume 1. Springer, 2004.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2712–2721. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/hendrycks19a.html`.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 2744–2751. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.acl-main.244. URL https://doi.org/10.18653/v1/2020.acl-main.244.

Jeremy Kawahara, Sara Daneshvar, Giuseppe Argenziano, and Ghassan Hamarneh. Seven-point checklist and skin lesion classification using multitask multimodal neural nets. *IEEE journal of biomedical and health informatics*, 23(2):538–546, 2018.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pp. 2668–2677. PMLR, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

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.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th international conference on computer vision*, pp. 365–372. IEEE, 2009.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 951–958. IEEE, 2009.

Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T Freeman, Phillip Isola, Amir Globerson, Michal Irani, et al. Explaining in style: Training a gan to explain a classifier in stylespace. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 693–702, 2021.

Weixin Liang and James Zou. Metashift: A dataset of datasets for evaluating contextual distribution shifts and training conflicts. *arXiv preprint arXiv:2202.06523*, 2022.

Adriano Lucieri, Muhammad Naseer Bajwa, Stephan Alexander Braun, Muhammad Imran Malik, Andreas Dengel, and Sheraz Ahmed. On interpretability of deep learning based skin lesion classifiers using concept activation vectors. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10. IEEE, 2020.

Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *arXiv preprint arXiv:2105.04289*, 2021.

Alexander Mathis, Thomas Biasi, Steffen Schneider, Mert Yuksekgonul, Byron Rogers, Matthias Bethge, and Mackenzie W Mathis. Pretraining boosts out-of-domain robustness for pose estimation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1859–1868, 2021.

Thomas McGrath, Andrei Kapishnikov, Nenad Tomašev, Adam Pearce, Demis Hassabis, Been Kim, Ulrich Paquet, and Vladimir Kramnik. Acquisition of chess knowledge in alphazero. *arXiv preprint arXiv:2111.09259*, 2021.

SW Menzies, C Ingvar, and WH McCarthy. A sensitivity and specificity analysis of the surface microscopy features of invasive melanoma. *Melanoma research*, 6(1):55–62, 1996.

Eric Mitchell, Charles Lin, Antoine Bosselut, Chelsea Finn, and Christopher D Manning. Fast model editing at scale. *arXiv preprint arXiv:2110.11309*, 2021.

osmr. imgclsmob: Deep learning networks. `https://github.com/osmr/imgclsmob`, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Veronica Rotemberg, Nicholas Kurtansky, Brigid Betz-Stablein, Liam Caffery, Emmanouil Chousakos, Noel Codella, Marc Combalia, Stephen Dusza, Pascale Guitera, David Gutman, et al. A patient-centric dataset of images and metadata for identifying melanomas using clinical context. *Scientific data*, 8(1):1–8, 2021.

Shibani Santurkar, Dimitris Tsipras, Mahalaxmi Elango, David Bau, Antonio Torralba, and Aleksander Madry. Editing a classifier by rewriting its prediction rules. *Advances in Neural Information Processing Systems*, 34, 2021.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. Editable neural networks. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=HJedXaEtvS`.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems*, 33:20554–20565, 2020.

A Related Works

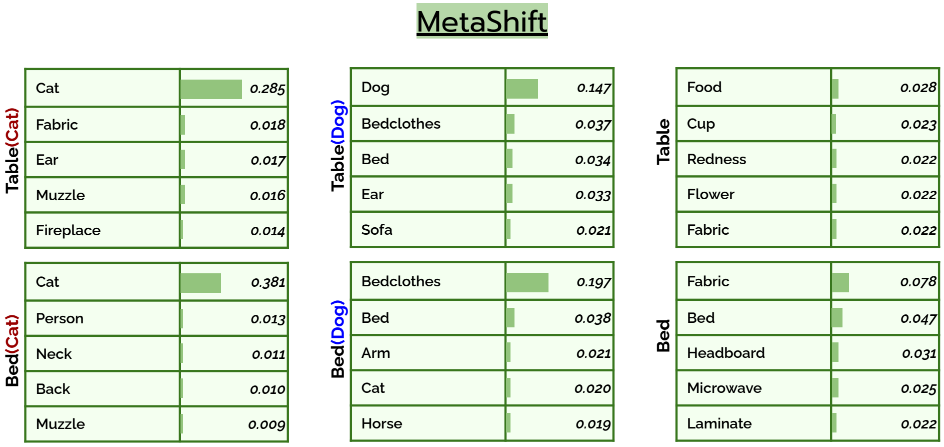
Concepts Human-understandable concepts draw increasing interest to interpret model behavior(Kim et al., 2018; Bau et al., 2017; 2020). Related work focuses on understanding if neural networks encode and use concepts (Lucieri et al., 2020; Kim et al., 2018; McGrath et al., 2021), or generate counterfactual explanations to understand model behavior(Ghandeharioun et al., 2021; Abid et al., 2021; Akula et al., 2020). These works mostly use a set of human-specified concepts to analyze model behavior, however, there is an increasing interest in automatically discovering the concepts that are used by a model (Yeh et al., 2020; Ghorbani et al., 2019; Lang et al., 2021).

Concept bottleneck models (CBMs) (Koh et al., 2020) extend the earlier idea (Lampert et al., 2009; Kumar et al., 2009) of decomposing a task into two parts by first predicting the concepts, then using concepts predicting the target. CBMs require training the model in an end-to-end fashion using concept labels at training time. Margeloiu et al. (2021) questions the interpretability of concepts in a CBM, and provide results that concepts learned by the CBMs may not be interpretable.

Model Editing Model editing aims to achieve the removal or modification of information in a given neural network. One branch of work focuses on intervening on the latent space of neural networks to alter the generated output towards a desired state, e.g. removal of artifacts or manipulation of object positions (Sinitsin et al., 2020; Bau et al., 2020; Santurkar et al., 2021). Bau et al. (2020) edits generative models, Santurkar et al. (2021) edits classifiers by modifying 'rules', such as making a model perceive the concept of a 'snowy road' as the concept of a 'road'. Mitchell et al. (2021); De Cao et al. (2021) aim to edit the factual knowledge in language models by training a separate network to modify model parameters achieving the desired edit.

B METASHIFT CONCEPT WEIGHTS

We observe that when a spurious correlation is present, the interpretable predictor in P-CBM's often assigns the largest weights to the concepts that were spuriously correlated with the target class. For instance, in the 5-class object recognition task *beach, computer, motorcycle, stove, table*, when examples of *table* images in our training set all contained a *cat*, the *cat* concept was assigned the most positive weight. A similar phenomenon occurs when *table* is spuriously correlated with *dog*. In contrast, *table* without spurious correlations had concept weights that appeared more generally related to *table*, e.g. *food*. With enough induced sparsity, i.e. with a large enough λ, there often emerges one or two concepts that have a substantively larger weight than the rest, making it easy to spot potential biases in the model.



Table(Cat)

Cat	0.285
Fabric	0.018
Ear	0.017
Muzzle	0.016
Fireplace	0.014

Table(Dog)

Dog	0.147
Bedclothes	0.037
Bed	0.034
Ear	0.033
Sofa	0.021

Table

Food	0.028
Cup	0.023
Redness	0.022
Flower	0.022
Fabric	0.022

Bed(Cat)

Cat	0.381
Person	0.013
Neck	0.011
Back	0.010
Muzzle	0.009

Bed(Dog)

Bedclothes	0.197
Bed	0.038
Arm	0.021
Cat	0.020
Horse	0.019

Bed

Fabric	0.078
Bed	0.047
Headboard	0.031
Microwave	0.025
Laminate	0.022

Figure 3: Examples of Metashift concept weights.

C METASHIFT EXPERIMENTS FOR MODEL EDITING

For Metashift, we have 2 tasks. Both tasks are 5-class object recognition tasks, where in the first one classes are *airplane, bed, car, cow, keyboard*, and for the second one we have *beach, computer, motorcycle, stove, table*. For each of these, we use a ResNet18 pretrained on ImageNet as the backbone of the P-CBM, and then use 100 images per class to train the concept bottleneck. For all experiments, we use the Adam Optimizer with a learning rate of 0.05, the regularization parameters $\lambda = 0.05$, $\alpha = 0.99$. Similar to CIFAR experiments, we use the Broden Concept dataset used in (Abid et al., 2021). Below we give the entire set of results.

Train	Test	Model	Original	Prune	Prune+Normalize	Fine-Tune
bed(dog)	bed(cat)	P-CBM	0.760	0.760	0.760	0.920
bed(cat)	bed(dog)	P-CBM	0.680	0.700	0.720	0.940
table(dog)	table(cat)	P-CBM	0.520	0.540	0.620	0.760
table(cat)	table(dog)	P-CBM	0.660	0.700	0.740	0.760
table(books)	table(dog)	P-CBM	0.600	0.580	0.780	0.720
table(books)	table(cat)	P-CBM	0.620	0.680	0.800	0.820
car(dog)	car(cat)	P-CBM	0.718	0.718	0.744	0.949
car(cat)	car(dog)	P-CBM	0.620	0.760	0.840	0.840
cow(dog)	cow(cat)	P-CBM	0.778	0.750	0.778	0.944
keyboard(dog)	keyboard(cat)	P-CBM	0.620	0.580	0.720	0.940
bed(dog)	bed(cat)	HP-CBM	0.760	0.760	0.780	0.900
bed(cat)	bed(dog)	HP-CBM	0.760	0.740	0.760	0.940
table(dog)	table(cat)	HP-CBM	0.600	0.620	0.640	0.780
table(cat)	table(dog)	HP-CBM	0.540	0.580	0.640	0.820
table(books)	table(dog)	HP-CBM	0.660	0.700	0.760	0.680
table(books)	table(cat)	HP-CBM	0.760	0.800	0.820	0.780
car(dog)	car(cat)	HP-CBM	0.795	0.769	0.795	0.974
car(cat)	car(dog)	HP-CBM	0.640	0.660	0.740	0.720
cow(dog)	cow(cat)	HP-CBM	0.639	0.639	0.639	0.917
keyboard(dog)	keyboard(cat)	HP-CBM	0.400	0.460	0.560	0.940

Table 3: Results of the Metashift editing experiments.