

Disentangling Polysemantic Channels in Convolutional Neural Networks

Robin Hesse¹ Jonas Fischer² Simone Schaub-Meyer^{1,3} Stefan Roth^{1,3}

¹Technical University of Darmstadt ²Max Planck Institute for Informatics ³hessian.AI

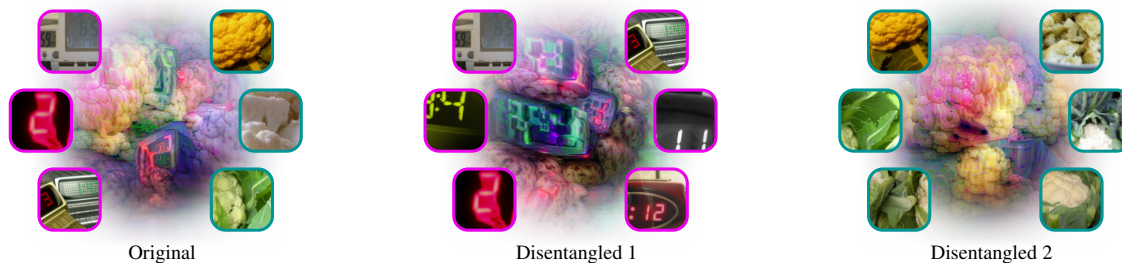


Figure 1. Strongly activating image patches and MACO [8] visualizations for channel #1660 of an ImageNet-trained ResNet-50 and the two corresponding disentangled channels (see Sec. 3.2). The original channel #1660 is polysemantic as it strongly activates for unrelated concepts occurring in “digital clock” and “cauliflower” images. After disentangling, we obtain channels that are significantly stronger activating *either* to concepts occurring in “digital clock” (Disentangled 1) *or* to concepts occurring in “cauliflower” (Disentangled 2).

Abstract

Mechanistic interpretability is concerned with analyzing individual components in a (convolutional) neural network (CNN) and how they form larger circuits representing decision mechanisms. These investigations are challenging since CNNs frequently learn polysemantic channels that encode distinct concepts, making them hard to interpret. To address this, we propose an algorithm to disentangle a specific kind of polysemantic channel into multiple channels, each responding to a single concept. Our approach restructures weights in a CNN, utilizing that different concepts within the same channel exhibit distinct activation patterns in the previous layer. By disentangling these polysemantic features, we enhance the interpretability of CNNs, ultimately improving explanatory techniques such as feature visualizations.¹

1. Introduction

Although convolutional neural networks (CNNs) are vital to many areas of computer vision, a major limitation is their increasing complexity, impeding their interpretability. As a result, the idea of *mechanistic interpretability* has been proposed, which focuses on understanding the information encoded in individual neurons or channels and how these form larger circuits that encode the decision mechanism of the network. In particular, feature visualization by optimization [7], *i.e.*, generating an image that maximizes the activation of a

CNN neuron/channel, is a widely-used method for visualizing encoded concepts. A common issue for such approaches is the presence of *polysemantic* neurons, resp. channels, that activate for multiple unrelated concepts. Those are less interpretable because post-hoc explanation methods like feature visualizations have not been designed to visualize multiple concepts simultaneously. Polysemanticity can occur because the CNN uses a non-privileged basis to encode concepts [6] or because the CNN learns to encode more concepts than channel dimensions by superimposing the concepts in the channel [6]. To resolve this issue, there have been efforts to understand the functioning of individual, polysemantic neurons [14–16, 21] or to identify *monosemantic* directions in feature space [12, 18]. We take a step further and aim to *disentangle* polysemantic channels. This allows moving back to analyzing *monosemantic* concepts learned by *individual* channels, which is more in line with the principles of mechanistic interpretability. As opposed to concurrent work [5], we propose an *explicit* disentanglement instead of a “virtual” one; our approach hence enables, *inter alia*, feature visualization through optimization (see Fig. 1). Specifically, our main contributions can be summarized as: (1) We propose a novel approach to identify polysemantic feature channels in CNNs. (2) Using the identified channels, we introduce a method to disentangle a polysemantic channel into multiple channels, only activating for one of the concepts each.

2. Related Work

Closest to our work is [5], which disentangles polysemantic neurons by assigning their active circuit to specific circuit

¹Code available at github.com/visinf/disentangle-channels.

clusters corresponding to each concept. While this “virtual” disentanglement enables “classifying” the active concept, it is not inherently part of the network, making standard XAI methods, such as feature visualization, non-trivial or even impossible. In contrast, we propose an *explicit* disentanglement (see Fig. 2) that allows direct application of established mechanistic interpretability tools. Another line of research visualizes the *different* concepts encoded in a *single* feature [14–16, 21]. E.g., [15] proposes an initialization scheme, allowing for the feature visualization of different concepts encoded in a polysemantic feature. Instead of focusing on individual features, [12, 18] aims to find meaningful *directions in feature space* that correspond to a *monosemantic* concept. Accordingly, a concept activation vector [12] is defined as the normal to the classification boundary in the feature space of the layer of interest, separating images that contain a concept of interest and images without that concept. Also closely related to our approach are concept whitening [2] and sparse autoencoders [11]. In the latter, a layer of interest is projected into a higher dimensional space while enforcing sparsity constraints to obtain a more monosemantic encoding. However, this approach is only an approximation of the original network. Further, it does not allow for an understanding of *individual* neurons and comes with its own set of challenges [10, 20, 24].

3. Disentangling Polysemantic Channels

We consider a classification setup with a set of images X belonging to one of n classes $t \in \{1 \dots n\}$. We let $f_{l_1}(x)_{i_1}$ denote the channel i_1 in the output of a CNN f at layer l_1 on input x . The relevance/attribution of an earlier channel i_2 in layer l_2 to the activation of a later channel i_1 in layer l_1 , $l_1 > l_2$ can be computed via

$$\mathcal{A}_{l_2, i_2}^{l_1, i_1}(x; f) = \text{sum} \left(f_{l_2}(x)_{i_2} \frac{\partial \text{sum}(f_{l_1}(x)_{i_1})}{\partial f_{l_2}(x)_{i_2}} \right), \quad (1)$$

where $\text{sum}(\cdot)$ denotes the sum along the spatial dimensions, aggregating a two-dimensional channel into a scalar (similar to global average pooling). Due to its simplicity, we here use Input×Gradient [22] to define $\mathcal{A}$, which we found to work sufficiently well. However, $\mathcal{A}$ could also rely on other attribution methods, such as LRP [1] or IG [23].

3.1. Identifying polysemantic channels

Finding candidate channels. We assume that a single channel is detecting one or more (abstract) concepts.² We consider channels as *potentially* polysemantic if the concepts they encode are occurring in different classes, i.e., if they are

²According to the principles of superposition and non-privileged bases [6], some concepts are encoded by a combination of channels. However, in the general sense, each channel encodes some information, which could also be regarded as a concept.

relevant to multiple classes. For each class index t and the corresponding training images $X^{(t)}$, we compute the relative attribution of the logit target output with respect to our layer of interest l and consider a channel as relevant if its relative attribution exceeds a threshold τ for a fraction p of $X^{(t)}$. Formally, a channel c in layer l is relevant for the t^{th} class if

$$\left(\frac{1}{|X^{(t)}|} \sum_{x \in X^{(t)}} \left[\frac{\mathcal{A}_{l, c}^{l^\omega, t}(x; f)}{\sum_{i \in \mathcal{C}_l} \mathcal{A}_{l, i}^{l^\omega, t}(x; f)} > \tau \right] \right) > p, \quad (2)$$

with $[\cdot]$ denoting the Iverson bracket [13], l^ω denoting the final layer, and $\mathcal{C}_l$ all channel indices in layer l . Channels relevant for two or more classes are considered *polysemanticity candidates*. This can happen because (1) a channel activates for a single concept shared by classes (e.g., different dog species), meaning no polysemanticity occurs or (2) a channel activates for multiple concepts, resulting in polysemanticity.

Identifying polysemanticity in candidate channels requires to distinguish between possibility (1) and (2). We build our approach on two observations: similar concepts should have similar activations not only in our layer of interest l , but also in the preceding layer $l-1$; and polysemanticity can be caused by superposition, which preferably occurs in sparse configurations, i.e., when superimposed concepts – e.g., a beak and a car mirror – rarely co-occur. From this, we infer that scenario (1) (no polysemanticity) occurs predominantly for a candidate channel relevant for classes t_1 and t_2 if the channel activations in the preceding layer $l-1$ relevant for c are *similar* for images from t_1 and t_2 . Vice versa, scenario (2) (polysemanticity) occurs predominantly if relevant channel activations in $l-1$ are *different* for images from t_1 and t_2 . To distinguish the two, we first compute the average relevance vectors (ARV $\in \mathbb{R}^{|\mathcal{C}_{l-1}|}$) in layer $l-1$

$$\text{ARV}(l, c, t; f) = \left(\frac{1}{|X^{(t)}|} \sum_{x \in X^{(t)}} \mathcal{A}_{l-1, i}^{l, c}(x; f) \right)_{i \in \mathcal{C}_{l-1}}, \quad (3)$$

with $\mathcal{C}_{l-1}$ being all channel indices in $l-1$. The ARV indicates the average relevance of each channel in layer $l-1$ to channel c in layer l for images of class t . With $S_{\cos}$ denoting the cosine similarity, we get the following definition.

Definition 3.1 (γ -polysemanticity). Given a channel c in layer l that was found to be relevant for target classes t_1, t_2 as per Ineq. (2), we say that c is affected by γ -polysemanticity for the concepts it detects in t_1 and t_2 w.r.t. a threshold γ iff

$$S_{\cos}(\text{ARV}(l, c, t_1; f), \text{ARV}(l, c, t_2; f)) < \gamma. \quad (4)$$

In the following, “polysemanticity” will refer to γ -polysemanticity. Polysemanticity of more than two concepts can be broken down into several two-concept cases; hence, we only consider disentanglement for the two-concept case.

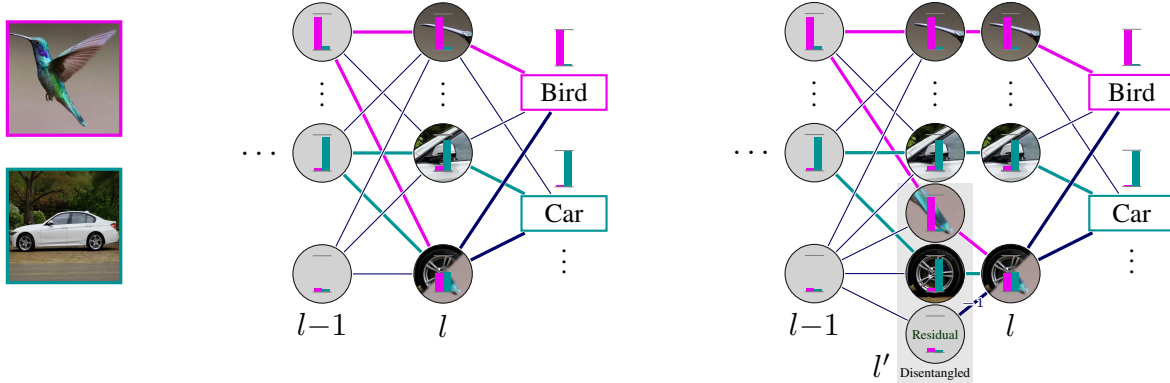


Figure 2. *Illustration of our proposed disentanglement approach.* We show two input samples from two different classes (*left*), the original neural network (*middle*), and our disentangled neural network (*right*). The bar plots in each channel indicate how active that channel is for each of the two input images. The color of the edge indicates if that edge is mainly propagating information from the first image, from the second image, or from both/none. The images in the nodes represent the encoded concepts.

3.2. Disentangling polysemantic channels

Building on the above identification of polysemantic channels, we propose an algorithm to disentangle them in any pre-trained CNN without affecting its output or predictive performance. Given a polysemantic channel c and the two classes t_1 and t_2 for which the channel is relevant, we aim to disentangle the channel into two channels, each of which only activates for one of the respective class concepts. In a nutshell, we create an artificial layer l' between layer $l-1$ and the layer l containing the channel we want to disentangle. The new layer l' will contain the disentangled channels, while layers $l-1$ and l remain unchanged. We outline the idea in Fig. 2. The modification of the network consists of three steps: (i) construction of the new layer l' , (ii) connection of layer l' with layer l , and (iii) connection of layer $l-1$ with the new layer l' .

Construction of layer l' starts by copying layer l . We then replace channel c in l' with three new channels; the first two channels will represent the two disentangled concepts while the third channel serves as a residual channel to ensure recovery of the original activations.

Connection of layer l' with l consists of simple identity mappings for all corresponding channels. The first two newly introduced channels are connected with an identity mapping to the channel c in l that we aim to disentangle. The third new channel is connected to c with a weight of -1 ; the reason for this follows below.

Connection of layer $l-1$ with l' consists of the previously existing connections from $l-1$ to l for all but the three new channels in l' . For the latter, we first set weights from $l-1$ to the weights originally going from $l-1$ to c . Then, for each channel i in $l-1$ we check whether it is relevant for the concept in t_2 using the average relevance

$$\text{ARV}(l, c, t_2; f)_i \cdot \rho > \text{ARV}(l, c, t_1; f)_i, \quad (5)$$

with $\rho \in [0, 1]$ controlling how much larger the relevance for t_2 needs to be than for t_1 in order for a channel i in $l-1$ to be considered only relevant for detecting the concept in t_2 . Please refer to Appendix A for an algorithm to automatically set ρ . We obtain the final weight for the first of the three new channels in l' by setting all edges coming from $l-1$ to zero where Ineq. (5) holds. Intuitively, we have now deleted the connections to channels in $l-1$ that contribute to detecting concept 2, and thus, this first new channel now mainly detects concept 1. For the second channel, we repeat the process, now removing weights responsible for concept 1, following an analogous procedure. The original channel has now been “disentangled” into these two channels. Any features beyond these two concepts are, however, now encoded twice – once in each of the two new channels. To address this issue, in the third channel, we delete all the edges coming from a channel in $l-1$ that is *either* strongly responsible for concept 1 *or* concept 2, *i.e.*, we only keep the edges that cannot be associated exclusively to concept 1 or 2. As the third channel is connected to the original channel c in layer l with a weight of -1 , it accounts for this double encoding and allows us to recover the original activation of channel c in layer l .

4. Experiments

The detailed experimental setup is provided in Appendix A. Briefly, we study the penultimate layer of a ResNet-50 [9] trained on the ImageNet dataset [3] and select $\gamma = 0.5$. We use the ImageNet training split for disentanglement and the validation split for evaluation.

Qualitative analysis. As an illustrative example, we found that channel #1660 is relevant for the classes “digital clock” and “cauliflower”. Their ARVs have a relatively low cosine similarity of 0.47 (see Theorem 3.1), thus the channel is affected by γ -polysemanticity. We disentangle channel

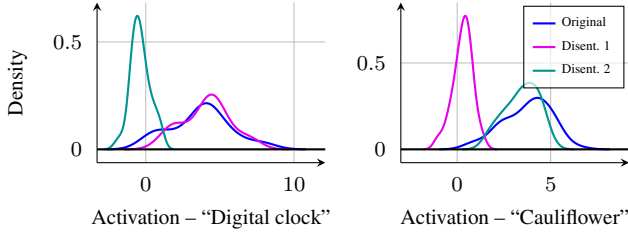


Figure 3. *Density plot of the activations for the original channel #1660 and the corresponding disentangled channels for images from the classes “digital clock” and “cauliflower”.* In both plots, one disentangled channel mimics the activation of the original channel while the other disentangled channel is mostly inactive, indicating that the disentanglement was successful.

#1660 to obtain two channels that detect the corresponding “digital clock” and “cauliflower” concepts, respectively (see Sec. 3.2). To verify the disentanglement, we feed evaluation images showing one of the two classes of interest through the original and disentangled model and plot the density function (kernel density estimate) for the activation of the original channel #1660 and the two disentangled channels in Fig. 3. Remarkably, the first disentangled channel mimics the activation of the original channel for the “digital clock” images while barely responding to “cauliflower” images, with the converse holding for the second channel, suggesting successful disentanglement.

To better understand the concepts a channel encodes in an image classification model, we provide images for which the channel activates the most and additionally use feature visualization by optimization [7]. However, visualizing a polysemantic channel presents two challenges: The visualization may highlight only one concept, partially obscuring the channel’s function, or it may capture both concepts together, making the explanation less interpretable [17]. Our disentangled representation of a channel alleviates this problem, which we exemplify by visualizing the concepts activating channel #1660 and the two disentangled channels, respectively, in Fig. 1. To visualize a feature by optimization, we use MACO [8]. For the original channel, we see that it is most strongly activated for images from “digital clock” and “cauliflower” alike, and even in the MACO visualization, we see that both cauliflower patterns and digital clock parts are visible. The disentangled channels only activate for images of one of the two classes, and in the MACO visualization, the respective concepts appear more isolated for the disentangled channels. We also observe that some cauliflower patterns are still lightly visible in the MACO visualization for the first disentangled channel. This might be due to the two disentangled channels still sharing various edges that have not been found to be particularly relevant for the concept in class 1, respectively, class 2, according to Ineq. (5).

Quantitative analysis. We next consider all polyse-

Concept	Disentangled 1	Disentangled 2	Residual
Concept 1	114 ↑	−9 ↓	0 ↓
Concept 2	−21 ↓	125 ↑	0 ↓

Table 1. *Quantitative evaluation.* We report the activation for the disentangled and residual channel relative to the activation of the original channel (in %) for images from the first (Concept 1) and second class (Concept 2).

mantic channels in the penultimate layer – affected by γ -polysemanticity with $\gamma = 0.5$ – for a quantitative analysis. For each such channel, we measure the mean relative activation after disentanglement – *i.e.*, the activation divided by the activation of the original (entangled) channel – of the corresponding disentangled and residual channels for evaluation images of the two classes of interest in Tab. 1. Confirming insights from the qualitative analysis, the first new channel almost exactly mimics the activation of the original channel (114%) for concept 1 images while only marginally (−9%) activating for concept 2 images. We see the opposite trend for the second new channel with −21% and 125% activation for concept 1 and concept 2, respectively. Moreover, the residual channel’s relative activation is very close to zero, suggesting it is only responsible for a minuscule amount of information. Thus, it does not necessarily need to be explained explicitly. These promising results indicate that our proposed algorithm for disentangling polysemantic channels is effective.

5. Conclusion & Limitations

We introduce a new mechanistic interpretability tool that identifies and disentangles channels affected by γ -polysemanticity into multiple channels, each only activating for one of the concepts. While not directly applicable to other forms of polysemanticity, our formal definition of γ -polysemanticity allows us to assess the effectiveness of our approach, both qualitatively and quantitatively. Currently only applied to the common form of disentanglement of *two* concepts, we anticipate that our method is easily extendable to more classes, *e.g.*, through a recursive application. In this initial work, we focused on a specific set of hyperparameters that gave reasonable results and aligned with previous knowledge (*e.g.*, association of ARV similarity with WordNet similarity; see Appendix A), yielding a proof of concept for our general approach. Future work could further explore the impact of these hyperparameters and potential methods for setting them automatically. Additionally, our current formulation of polysemanticity is based on class labels, thus, our proposed disentanglement may work better in later layers with more semantically meaningful concepts. Although preliminary, we are confident that our work will be a valuable extension of the current landscape of mechanistic interpretability.

Acknowledgments. This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme (grant agreement No. 866008). The project has also been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – project number 529680848. Further, the project has been supported by the State of Hesse, through the cluster projects “The Third Wave of Artificial Intelligence (3AI)” and “The Adaptive Mind (TAM),” and by the Hessian Ministry of Higher Education, Research, Science and the Arts and its LOEWE research priority program “WhiteBox” under grant LOEWE/2/13/519/03/06.001(0010)/77.

References

- [1] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):1–46, 2015. 2
- [2] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nat. Mach. Intell.*, 2(12):772–782, 2020. 2
- [3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, pages 248–255, 2009. 3, 6
- [4] Thomas Deselaers and Vittorio Ferrari. Visual and semantic similarity in ImageNet. In *CVPR*, pages 1777–1784, 2011. 6
- [5] Maximilian Dreyer, Erblina Puelku, Johanna Vielhaben, Wojciech Samek, and Sebastian Lapuschkin. PURE: Turning polysemantic neurons into pure features by identifying relevant circuits. In *CVPR XAI4CV Workshop*, pages 8212–8217, 2024. 1
- [6] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022. 1, 2
- [7] Dumitru Erhan, Y. Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *Technical Report, Université de Montréal*, 2009. 1, 4
- [8] Thomas Fel, Thibaut Boissin, Victor Boutin, Agustin Picard, Paul Novello, Julien Colin, Drew Linsley, Tom Rousseau, Rémi Cadène, Lore Goetschalckx, Laurent Gardes, and Thomas Serre. Unlocking feature visualization for deep network with magnitude constrained optimization. In *NeurIPS*, pages 37813–37826, 2023. 1, 4
- [9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 3, 6
- [10] Thomas Heap, Tim Lawson, Lucy Farnik, and Laurence Aitchison. Sparse autoencoders can interpret randomly initialized transformers. *arXiv:2501.17727 [cs.LG]*, 2025. 2
- [11] Robert Huben, Hoagy Cunningham, Logan Riggs, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *ICLR*, 2024. 2
- [12] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie J. Cai, James Wexler, Fernanda B. Viégas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (TCAV). In *ICML*, pages 2673–2682, 2018. 1, 2
- [13] Donald E. Knuth. Two notes on notation. *The American Mathematical Monthly*, 99(5):403–422, 1992. 2
- [14] Jesse Mu and Jacob Andreas. Compositional explanations of neurons. In *NeurIPS*, pages 17153–17163, 2020. 1, 2
- [15] Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Multi-faceted feature visualization: Uncovering the different types of features learned by each neuron in deep neural networks. *arXiv:1602.03616 [cs.NE]*, 2016. 2
- [16] Tuomas P. Oikarinen and Tsui-Wei Weng. Linear explanations for individual neurons. *arXiv:2405.06855 [cs.LG]*, 2024. 1, 2
- [17] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. 4
- [18] Laura O’Mahony, Vincent Andrearczyk, Henning Müller, and Mara Graziani. Disentangling neuron representations with concept vectors. In *CVPRW*, pages 3770–3775, 2023. 1, 2
- [19] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. 6
- [20] Gonçalo Paulo and Nora Belrose. Sparse autoencoders trained on the same data learn different features. *arXiv:2501.16615 [cs.LG]*, 2025. 2
- [21] Biagio La Rosa, Leilani Gilpin, and Roberto Capobianco. Towards a fuller understanding of neurons with clustered compositional explanations. In *NeurIPS*, pages 70333–70354, 2023. 1, 2
- [22] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *ICML*, pages 3145–3153, 2017. 2
- [23] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328, 2017. 2
- [24] Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. AxBench: Steering LLMs? Even simple baselines outperform sparse autoencoders. *arXiv:2501.17148 [cs.CL]*, 2025. 2