AttributionLab: Faithfulness of Feature Attribution Under Controllable Environments

Yang Zhang[1*] **Yawei Li**[2,3*] **Hannah Brown**[1]
Mina Rezaei[2,3] **Bernd Bischl**[2,3] **Philip Torr**[4] **Ashkan Khakzar**[4]
Kenji Kawaguchi[1]
[1] National University of Singapore
[2] LMU Munich [3] Munich Center for Machine Learning
[4] University of Oxford
yangzhang@u.nus.edu kenji@comp.nus.edu.sg
{yawei.li, bernd.bischl, mina.rezaei}@stat.uni-muenchen.de
{ashkan.khakzar, philip.torr}@eng.ox.ac.uk

Abstract

Feature attribution explains neural network outputs by identifying relevant input features. How do we know if the identified features are indeed relevant to the network? This notion is referred to as *faithfulness*, an essential property that reflects the alignment between the identified (attributed) features and the features used by the model. One recent trend to test faithfulness is to design the data such that we know which input features are relevant to the label and then train a model on the designed data. Subsequently, the identified features are evaluated by comparing them with these designed ground truth features. However, this idea has the underlying assumption that the neural network learns to use *all* and *only* these designed features, while there is no guarantee that the learning process trains the network in this way. In this paper, we solve this missing link by *explicitly designing the neural network* by manually setting its weights, along with *designing data*, so we know precisely which input features in the dataset are relevant to the designed network. Thus, we can test faithfulness in *AttributionLab*, our designed synthetic environment, which serves as a sanity check and is effective in filtering out attribution methods. If an attribution method is not faithful in a simple controlled environment, it can be unreliable in more complex scenarios. Furthermore, the AttributionLab environment serves as a laboratory for controlled experiments through which we can study feature attribution methods, identify issues, and suggest potential improvements.

1 Introduction

Neural networks exhibit increasing capabilities as the scale of their design and their training data increases [8, 40, 22, 21, 9, 7]. These capabilities are achieved through the use of basic architectural blocks [32, 14, 38]. Though we know the architectural design of these networks and know their computational graph explicitly, we do not have a human interpretable understanding of neural networks. One way to explain the neural network output is to *identify important input features* for a single prediction, an explanation paradigm known as *input feature attribution*. There has been an ongoing quest for finding attribution methods to explain neural network functions [28, 41, 5, 34, 30, 13, 19, 42, 37, 27]. However, one challenge remains: How can we know whether the features identified by attribution are aligned with features relevant to the

[*]Equal contribution

XAI in Action: Past, Present, and Future Applications @ NeurIPS 2023.

neural network? I.e., how do we know if an attribution is *faithful*? An attribution may seem reasonable to us, but the neural network may use other input features. Conversely, an attribution may seem unreasonable but be faithful and indeed reflect the features relevant to the neural networks. Moreover, in the presence of multiple differing attribution explanations [17, 16], it is unclear which explanation to trust. The increasing complexity of networks, compounded by the rising complexity of feature attribution methodologies, further convolutes the problem.

One recent trend to assess the faithfulness of attribution methods is through the use of synthetic data. Synthetic data are designed such that associations between features and labels are known to users [4, 2, 43]. *However, as we discuss in Section 3 there is no guarantee that the learning process will train the network to use the designed associations in the dataset*. Hence, the association learned by models can differ from the designed association in the synthetic dataset. Consequently, evaluation based on the designed association is not guaranteed to reflect the faithfulness of feature attribution methods. Furthermore, many evaluations usually report a performance score without any information on why an attribution method has limited performance. A test environment capable of uncovering properties and issues of methods can better contribute to the development of feature attribution research.

In scientific experiments with complex setups and multiple variables, it is typical to use a laboratory setting to conduct controlled tests. Analogously, our work proposes a paradigm for providing a controlled laboratory environment. In this laboratory environment, *both* the neural networks and the datasets are designed such that *we know which features are relevant* to the network output. Thus, we obtain the *ground truth attribution* in this synthetic environment. We leverage this information for the faithfulness test by measuring the alignment between the ground truth attribution and attribution maps (Section 4). A controlled environment can also be used to study the behavior of attribution methods under various circumstances by adjusting or ablating variables to simulate different scenarios. With the help of proposed synthetic environments, we examine a broad range of attribution methods and investigate the impact of several crucial factors, including the choice of baseline and superpixel segmentation (Section 5). We make several observations from the test results and provide suggestions for improving their attribution performance.

2 Related work

Since the advent of deep neural networks [18, 32, 14], understanding how these complex models make predictions came under the spotlight [31]. One approach is a simplified view of identifying features relevant to a prediction, known as **feature attribution**. Initial efforts to perform feature attribution focused on linear models [31, 5] and backpropagation [34, 41]. Subsequently, more principled approaches emerged, such as backpropagating output differences with respect to a reference input [5, 30] and axiomatic methods [37, 19] inspired by the Shapley value [29]. Meanwhile, intuitive approaches probing internal states [28, 27, 10] and optimizing input masks were introduced [13, 42]. With these advancements, **feature attribution evaluation** became a central question. Literature has proposed *sanity checks* testing whether attribution changes upon randomizing the network's weights [1] or if it is used on a different output class [33]. It also evaluates *faithfulness* by analyzing networks's output when perturbing input features based on their relevance [25, 3, 20, 15, 24], or show theoretically if they are aligned with axioms outlining desirable properties [37, 19] A recent trend in evaluating *faithfulness* is designing datasets with known input-feature-output associations, enabling comparison between attribution and ground truth. [4] generates datasets of shapes, [2] generates synthetic graph datasets, and [43] proposes a dataset modification procedure to incorporate ground truth. However, these works do not guarantee that the network uses the intended associations. To address this problem, [16] proposes a post-hoc solution of inverse feature generation by generating input features using the network.

3 A controllable environment to evaluate faithfulness

We aim to establish an environment where we know which input features are relevant to the output of a model. This laboratory setup allows for testing the faithfulness of feature attribution methods and understanding the sources of failures. Prior to detailing the setup, we underline the necessity of designing both the *data* and the *network* to obtain the ground truth attribution.

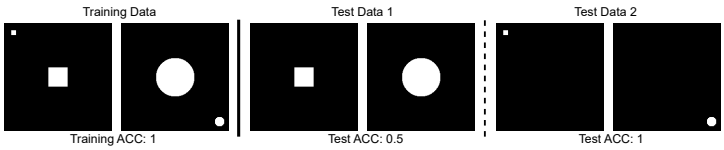


Figure 1: **Designing data is not enough.** Example on the neural networks not learning the designated ground truth features in the synthetic dataset. In this example, designed ground truth features are both objects in the center and on the edge. Even though the model can achieve 100% accuracy, our test shows that the model only learns to use designed features at the corner and ignore the central ground truth features.

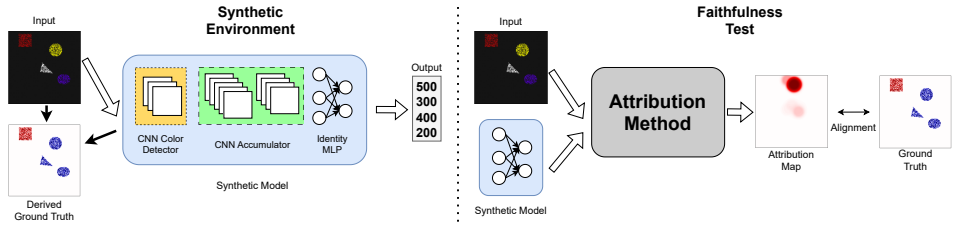


Figure 2: **Designing data and model** to set up a controllable environment for testing the faithfulness of attribution methods and analyzing their properties. To obtain the ground truth attribution, we explicitly design networks in tandem with inputs. The models follow conventional neural network designs and have sufficient complexity. The faithfulness test compares attribution results in the synthetic setting with the ground truth attribution.

To realize the controlled setup for evaluating explanations, one might design a dataset where the associations between input features and output labels are known: e.g., a simple dataset of squares and circles. Indeed, prior approaches [4, 43, 26, 2] design synthetic datasets and train a model on them. The underlying assumption is that a model learns the intended association in the dataset once the model achieves peak performance on the synthetic dataset during the learning process. However, it is unclear whether the trained network uses the entire square, straight lines, or just corners to identify the square. How can we know if a network uses an entire square for decision-making in this setup? This phenomenon is formulated as the *Rashomon effect*, which we rephrase in the following definition:

Rashomon effect [6] *There exist many models under the same hypothesis class that can achieve equally good accuracy but use different information for inference.*

The Rashomon effect states that it is generally invalid to assume that a model with 100% accuracy on a dataset $\mathcal{D}$ is ensured to learn the true labeling process. For instance, the model can learn other associations (e.g. use corners to classify a square). We provide empirical evidence for trained neural networks ignoring designed features. The result in Figure 1 shows a neural network, which achieves 100% training accuracy but learns to solely use partial ground truth features designed for the dataset (objects at the edge). In a nutshell, a model can learn to perform correctly on the dataset but is not guaranteed to learn the intended ground truth features in the data. *Therefore, we manually design the neural network*.

4 Design of data and neural network

We propose a modular setup where each component performs specific tasks and fulfills a purpose relevant to evaluating attribution methods. For the designs to facilitate the evaluation of feature attribution, we follow certain design principles. Firstly, the designs resemble real scenarios, such as image classification using a convolutional neural network (Figure 2). Designs that are similar to real-world cases can narrow the gap between real environments and synthetic environments. Hence, we can leverage synthetic environments to identify issues within attribution methods that are relevant to actual use cases. Furthermore, the design ensures that every ground-truth pixel is relevant and equally relevant. Specifically, with designed input data comprised of ground-truth (foreground) and baseline (background) pixels, designed neural networks have the following

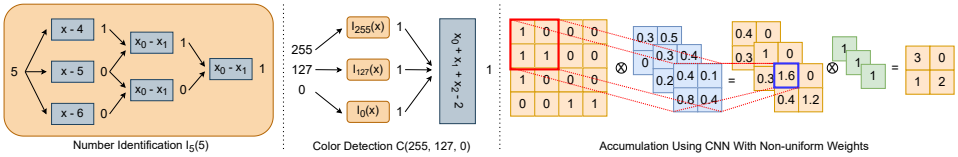


Figure 3: **Computational graph illustration** of our designed neural network modules. The left example shows a neural network of identifying number 5, and the middle example shows a simple color detector for detecting RGB value $(255, 127, 0)$. In these two cases, blue boxes symbolize neurons, with their respective computations indicated within the box. ReLU activation is applied after each neuron, which is omitted in the figure. The right example demonstrates CNN operations to achieve accumulation using non-uniform kernel weights.

properties: (1) *Sensitivity* property: The addition/removal of *any* ground-truth pixel to/from the background affects the output of the designed neural network. (2) *Symmetry* property: The addition/removal of any ground-truth pixel to/from the background *equally* affects the output of the designed neural network. These properties are aligned with the sensitivity and symmetry feature attribution axioms [36, 37, 19], which axiomatically define the relevance of a feature.

We design an environment that resembles real image classification tasks. The task is to identify the dominant color (in terms of number of pixels) within an input image (see Figure 2). The designed dataset comprises color images, each containing N_C patches of uniquely colored foreground objects and a distinct background color that is consistent across images. By treating each foreground color as a class, we formulate a multi-class task to predict the dominant color in an image. The designed classification model outputs softmax scores using logits for N_C classes, where the logit of each class corresponds to the sum of pixels of the respective color.

Simulating "Unseen Data Effect": It is common that trained neural networks have unintended outputs given inputs that are not from the training distribution. However, since the data and every module (and weights) are set up manually, we know the expected behavior of the network for any input. Specifically, the network exclusively counts the number of pixels of predetermined colors, and other input values (colors) do not affect the network output. Through a neural operation (explained below), we can upgrade the setup to simulate the behavior of trained neural networks given data not previously seen by the model during training.

The network is designed to perform the designated task and exhibit unpredictable behavior when given inputs that are not predetermined within the design (hence simulating the "Unseen Data Effect"). The first component of the network is a CNN color detector responsible for detecting target colors and simulating Unseen Data Effects. Its output has dimensions of $(N_C + N_R) \times H \times W$, where N_C denotes the number of target classes and N_R denotes the number of *redundant* channels (the redundant channels are the neural implementation of "Unseen Data Effect" simulation). For the first N_C channels, the i^{th} output map is the activation of the i^{th} target color. Firstly, we design a neural structure that can identify a specific integer number. For integer input, this structure can be defined as $I_N(x) = \mathrm{ReLU}(I_{>N-1}(x) - I_{>N}(x))$, where $I_{>i}(x) = \mathrm{ReLU}(\mathrm{ReLU}(x - i) - \mathrm{ReLU}(x - i - 1))$. Given a color to be detected that is in RGB format (R, G, B), where R, G, and B are integers within $[0, 255]$. For a pixel with intensity (r, g, b), the color detection mechanism shown in Figure 3 is $C(r, g, b) = \mathrm{ReLU}(I_R(r) + I_G(g) + I_B(b) - 2)$. Here, the number identification functions I_R, I_G, and I_B each detect a predefined component of an RGB value and are implemented as shown above. Hence, we have $C(r, g, b) = 1$ for $r = R, g = G, b = B$, and $C(r, g, b) = 0$ otherwise. The remaining N_R redundant channels activate on any other colors not among the $N_C + 1$ colors defined in our dataset. Specifically, if any pixel has a color that is neither a target color nor the background color, all N_R redundant channels will activate at the position of this pixel. The activation mechanism of redundant channels is implemented as $R(r, g, b) = \mathrm{ReLU}(-\sum C_i(r, g, b) + 1)$. Consequently, $R(r, g, b) = 1$ if all $C_i(r, g, b) = 0$, and $R(r, g, b) = 1$ if $C_i(r, g, b) = 1$ for any i. Following the color detector, we have a CNN module that accumulates activation of the first N_C channels respectively. Figure 3 illustrates the working principle of pixel accumulation using CNN with non-uniform weights. The remaining N_R redundant channels have random connections to the input of this CNN module. Therefore, the CNN accumulation module will have unexpected outputs if the input images contain any color that is not seen in the training dataset. Lastly, to preserve the conventional

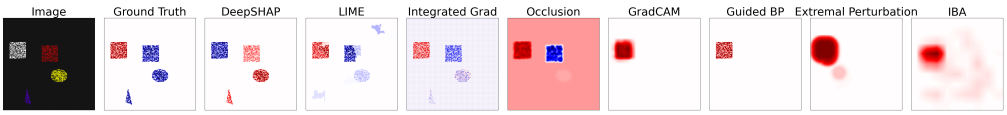
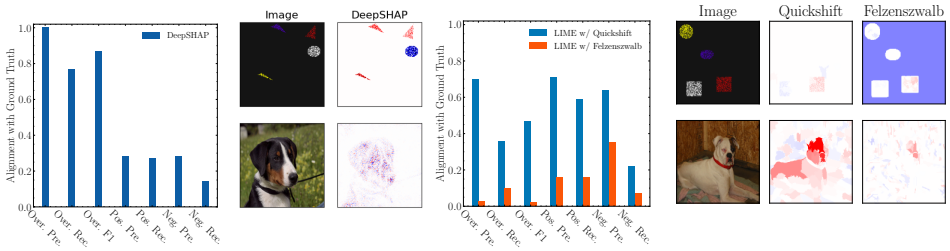


Figure 4: **Attributions in AttributionLab**. Red and blue colors denote positive and negative attribution to the target class, respectively. The sample is randomly selected.



(a) (b) (c) (d)

Figure 5: (a) Faithfulness test of **DeepSHAP** in AttributionLab. (b) Visual example of DeepSHAP. According to (a) and (b), DeepSHAP attributes correctly but fails to determine the positive/negative contribution. (c) Test result of **LIME** in AttributionLab. (d) Visual example of LIME. (c) and (d) show that the performance of LIME strongly depends on the segmentation.

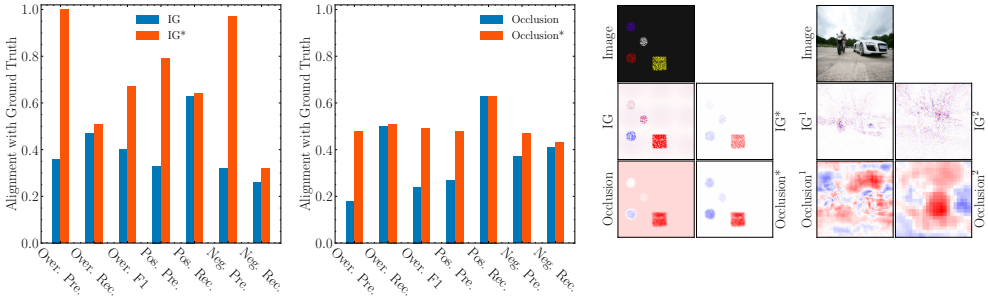
model architecture, we have an MLP that performs identity mapping, as the output of the CNN module already provides the logits of each target color for normal inputs.

5 Faithfulness test of attribution methods in AttributionLab

In this section, we deploy the designed environment to test the faithfulness of attribution methods and analyze their various aspects. To provide an intuition about the behavior of various attribution methods, we visualize them in AttributionLab in Figure 4. To quantitatively test the alignment between attribution maps and ground truth masks, we use precision, recall, and F_1-score. In multi-class classification, signed ground truth information is available, outlining the positive and negative contributions of features to the target class. To test attribution methods with signed ground truth, we separately compute the precision, recall, and F_1-score of the positive and negative portions of the attribution maps. In addition, we test the entire attribution map using an unsigned union of both positive and negative ground truth: the *overall ground truth*. This test takes into consideration all features that contribute to decision-making, ignoring the sign of attribution values. Furthermore, we employ these attribution methods on an ImageNet [11]-pretrained VGG16 [32] and check if the test result can generalize to the real world.

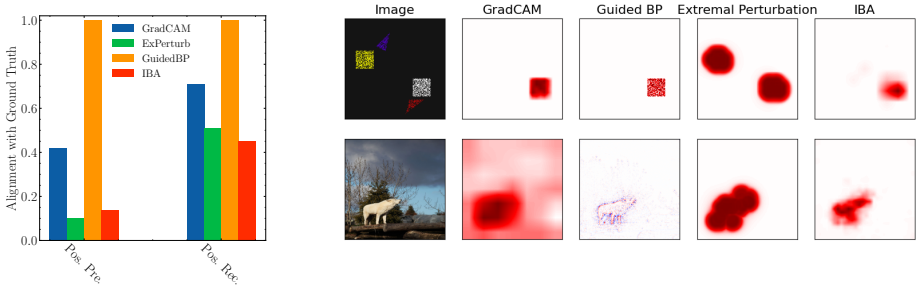
Faithfulness test of DeepSHAP. We test the faithfulness of DeepSHAP [19] by comparing the positive attribution with positive ground truth features (GT), negative attribution with negative GT, and overall attribution with overall GT by only considering the magnitude of attribution values. Figure 5a shows the test result of DeepSHAP in the synthetic environment. The overall precision, recall, and F_1-score reveal that DeepSHAP performs well in locating the contributing features when we disregard the sign of attribution. However, the low precision and recall of positive and negative attribution suggest that DeepSHAP encounters difficulty in discerning whether a feature contributes positively or negatively to the target class. Figure 5b further corroborated this issue in both synthetic and real-world scenarios. In the ImageNet example shown in the image, we observe both positive and negative attribution on pixels that are closely located and have similar colors. Our results suggest that DeepSHAP can be used to identify all relevant features for the model without considering the sign of attribution values.

Faithfulness test of LIME. LIME [23] requires the input image to be segmented into superpixels, it then treats all pixels within a superpixel as a single feature. Consequently, the resulting attribution map can be influenced by the segmentation step. To investigate the impact of segmentation, we utilize both Quickshift [39] and Felzenszwalb [12] segmentation algorithms and test the faithfulness of the resulting attributions. Figure 5c reveals noticeable difference between



(a) IG vs. IG* (b) Occ. vs. Occ.* (c) Syn. example (d) Real example

Figure 6: **IG and Occlusion with diverse baselines**. (a-b) IG* and Occlusion* employing the ground truth baseline display substantial enhancement in faithfulness test. (c) Comparative visualization of attribution maps, created with and without the utilization of the ground truth baseline. (d) Attribution maps generated on ImageNet. The superscripts signify the use of distinct baselines. The attributions highlight different areas when employing different baselines.



(a) Faithfulness test (b) Visual examples in synthetic setting and ImageNet

Figure 7: **Faithfulness test and visual examples** for GradCAM, GuidedBP, ExPerturb, and IBA. Some effectively identify positively contributing features, but none successfully discern negatively contributing features, inherent to their design limitations. GradCAM and IBA exhibit blurriness upon resizing, and the performance of ExPerturb is markedly influenced by the perturbation area.

the outcomes derived from these two segmentation techniques. Figure 5d provides additional visual evidence of these differences in real-world scenarios. In LIME, accurate attribution requires a finer segmentation process. Hence, LIME requires prior knowledge regarding the ground truth features, that is, which pixels belong together and which ones are independent.

Faithfulness test of Integrated Gradients (IG), and Occlusion.

Previous research [35] has revealed that IG [37] is sensitive to the choice of baseline. While testing IG with a proper baseline remains a challenge in the real world, our controlled experimental setup provides the unique advantage of access to the true baseline. We introduce this baseline-accurate variant of IG as IG*. Figure 6a reveals a significant enhancement in the precision of IG*. We see that sensitivity to baseline is not an ignorable issue. Analogous to IG*, we introduce Occlusion* that employs the true baseline. Figure 6b also demonstrates a notably improved precision of Occlusion*. Figure 6c and Figure 6d further illustrate the sensitivity of these methods to the choice of baseline. Based on these empirical findings, we underscore that one potential direction for enhancing these methods is the determination of an accurate baseline.

Faithfulness test of GradCAM, GuidedBP, ExPerturb, IBA. GradCAM [28], ExPerturb [13], IBA [27], and GuidedBP [34] do not identify pixels that negatively contribute to the target class, as evidenced by Figure 4 and Figure 7. This is because these methods typically initiate backpropagation from the logits before the softmax. Additionally, GradCAM, ExPerturb, and IBA generate blurry attribution maps, which cause them to have higher recall than precision.

6 Conclusion

In this work, we propose a controlled laboratory setup for testing feature attribution explanations. The crux of our approach is the implementation of a paired design, i.e., manually programming the neural network and designing the dataset. We test feature attribution explanations by simulating various conditions such as inappropriate baseline values for attribution methods and different segmentation masks as input of attribution methods, demonstrating their significant impact on attribution performance. Our proposed synthetic environment can empower future research by identifying potential failure modes of attribution methods in a trustable, controlled environment.

References

[1] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

[2] C. Agarwal, O. Queen, H. Lakkaraju, and M. Zitnik. Evaluating explainability for graph neural networks. *Scientific Data*, 10(1):144, 2023.

[3] M. Ancona, E. Ceolini, C. Öztireli, and M. Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. In *International Conference on Learning Representations*, 2018.

[4] L. Arras, A. Osman, and W. Samek. Clevr-xai: A benchmark dataset for the ground truth evaluation of neural network explanations. *Information Fusion*, 81:14–40, 2022.

[5] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7), 2015.

[6] L. Breiman. Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). *Statistical Science*, 16(3):199 – 231, 2001.

[7] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners, 2020.

[8] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

[9] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[10] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 839–847, 2018.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[12] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59:167–181, 2004.

[13] R. Fong, M. Patrick, and A. Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2950–2958, 2019.

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

[15] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim. A benchmark for interpretability methods in deep neural networks. *Advances in neural information processing systems*, 32, 2019.

[16] A. Khakzar, P. Khorsandi, R. Nobahari, and N. Navab. Do explanations explain? model knows best. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10244–10253, 2022.

[17] S. Krishna, T. Han, A. Gu, J. Pombra, S. Jabbari, S. Wu, and H. Lakkaraju. The disagreement problem in explainable machine learning: A practitioner's perspective, 2022.

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, 2012.

[19] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[20] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digital signal processing*, 73:1–15, 2018.

[21] OpenAI. Gpt-4 technical report, 2023.

[22] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

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

[24] Y. Rong, T. Leemann, V. Borisov, G. Kasneci, and E. Kasneci. A consistent and efficient evaluation strategy for attribution methods. In *International Conference on Machine Learning*, pages 18770–18795. PMLR, 2022.

[25] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2016.

[26] M. Schuessler, P. Weiß, and L. Sixt. Two4two: Evaluating interpretable machine learning - a synthetic dataset for controlled experiments, 2021.

[27] K. Schulz, L. Sixt, F. Tombari, and T. Landgraf. Restricting the flow: Information bottlenecks for attribution. In *International Conference on Learning Representations*, 2020.

[28] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[29] L. S. Shapley. A value for n-person games. *Contributions to the Theory of Games*, 2(28):307–317, 1953.

[30] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017.

[31] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations (ICLR 2015)*, pages 1–14, 2015.

[33] L. Sixt, M. Granz, and T. Landgraf. When explanations lie: Why many modified bp attributions fail. In *International Conference on Machine Learning*, pages 9046–9057. PMLR, 2020.

[34] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*, 2015.

[35] P. Sturmfels, S. Lundberg, and S.-I. Lee. Visualizing the impact of feature attribution baselines. *Distill*, 2020. https://distill.pub/2020/attribution-baselines.

[36] M. Sundararajan and A. Najmi. The many shapley values for model explanation. *37th International Conference on Machine Learning, ICML 2020*, 2020.

[37] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[39] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *Computer Vision–ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part IV 10*, pages 705–718. Springer, 2008.

[40] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

[41] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pages 818–833. Springer, 2014.

[42] Y. Zhang, A. Khakzar, Y. Li, A. Farshad, S. T. Kim, and N. Navab. Fine-grained neural network explanation by identifying input features with predictive information. *Advances in Neural Information Processing Systems*, 34:20040–20051, 2021.

[43] Y. Zhou, S. Booth, M. T. Ribeiro, and J. Shah. Do feature attribution methods correctly attribute features? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9623–9633, 2022.