Geometric Remove-and-Retrain (GOAR): Coordinate-Invariant eXplainable AI Assessment

Yong-Hyun Park[1], **Junghoon Seo**[2], **Bomseok Park**[1], **Seongsu Lee**[1], **Junghyo Jo**[1]
[1]Seoul Nataional University [2]SI-Analytics

Abstract

Identifying the relevant input features that have a critical influence on the output results is indispensable for the development of explainable artificial intelligence (XAI). Remove-and-Retrain (ROAR) is a widely accepted approach for assessing the importance of individual pixels by measuring changes in accuracy following their removal and subsequent retraining of the modified dataset. However, we uncover notable limitations in pixel-perturbation strategies. When viewed from a geometric perspective, this method perturbs pixels by moving each sample in the pixel-basis direction. However, we have found that this approach is coordinate-dependent and fails to discriminate between differences among features, thereby compromising the reliability of the evaluation. To address this challenge, we introduce an alternative feature-perturbation approach named Geometric Remove-and-Retrain (GOAR). GOAR offers a perturbation strategy that takes into account the geometric structure of the dataset, providing a coordinate-independent metric for accurate feature comparison. Through a series of experiments with both synthetic and real datasets, we substantiate that GOAR's geometric metric transcends the limitations of pixel-centric metrics.

1 Introduction

Feature attribution methods, which involve the identification of input features most relevant to a model's output, are pivotal in explainable artificial intelligence (XAI) research [30, 29, 31, 38, 25]. These explanations have been used to both analyze [19, 20, 43] and debug [2, 1] neural networks. As various attribution methods are being proposed, it becomes essential to set up benchmarks to evaluate them [23, 9, 41, 11].

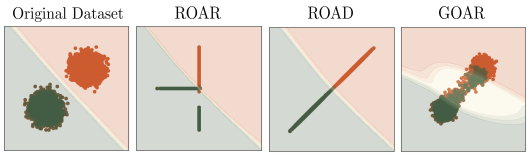

Figure 1: **Pitfalls of pixel-perturbation approaches.** The shaded regions represent the decision boundaries of models trained on each dataset. Pixel-perturbation methods like Remove-and-Retrain (ROAR) and Remove-and-Debias (ROAD) fail to eliminate information from the dataset, which results in little degradation in performance even after removing features. In contrast, our GOAR offers a geometrically natural way to remove features. It enables a precise and effective comparison of attribution methods.

A widely used approach for assessing attribution methods is the *pixel-perturbation* strategy. This strategy removes pixels associated with relevant features from an image and then measures the model performance after retraining on this modified dataset. If the model's accuracy drops significantly, it indicates the method successfully identified crucial features. Within this strategy, two prominent metrics stand out: "Remove-and-Retrain" (ROAR) [11], which replaces important pixels with a fixed value, and "Remove-and-Debias" (ROAD) [25], which substitues pixels with a noisy linear interpolation.

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

Despite the widespread use of pixel-perturbation strategies, we reveal a significant geometric issue associated with them.To elucidate this issue, we consider a 2-class Gaussian mixture dataset (Figure 1). As the perturbation strength of input increases, one can observe how identified features impact the output classification. In this example, both ROAR and ROAD fail to effectively erase information, leaving the dataset still separable. Neither good nor bad features cannot lead to an accuracy drop after pixel-perturbation, resulting in ineffective comparison of features. Our analysis indicates that this problem originates from the reliance of feature removal on pixel coordinates which are unrelated to the intrinsic geometric structure of the dataset.

To address this problem, we introduce a *feature-perturbation* strategy named Geometric Remove-and-Retrain (GOAR). GOAR perturbs images by moving the data sample along the feature direction, eliminating the dependence on pixel coordinates. However, naïvely shifting the image along the feature direction can lead to a retrained model predicting class labels based on spurious patterns related to the subtracted features. We tackle this issue by leveraging the diffusion model to ensure that perturbed data points remain closely aligned with their original data manifold. Unlike other metrics, GOAR removes information from the dataset in a precise and effective way, as shown in Figure 1.

We validate our geometric analysis and the effectiveness of GOAR through experiments on both synthetic and real datasets. In a toy example, we show that pixel-perturbation strategies have problems in comparing features, whereas GOAR provides an accurate comparison. Furthermore, we conduct experiments on MNIST [6], FashionMNIST [40], and CIFAR10 [15] to show that pixel-perturbation strategies have drawbacks in comparing various attribution methods, whereas GOAR appropriately compares these methods without such problems.

Our contributions are summarized as follows:

- We examine pixel-perturbation strategies from a geometric perspective and reveal their limitations arising from their reliance on pixel coordinates.
- We propose a new feature-perturbation strategy named GOAR.
- We validate our geometric analysis and demonstrate the efficacy of GOAR through experiments on both synthetic and real datasets.

2 Preliminary

Remove-and-Retrain (ROAR) [11]. ROAR operates by removing pixels associated with important features. The performance is then evaluated based on the drop in accuracy when the model is retrained on this modified dataset. A steeper decline in performance suggests that the attribution method has more accurately identified the critical features upon which the model relies for predictions. This concept is referred to as *fidelity* [8].

To be self-contained, we define the ROAR process as follows: Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, where $x \in \mathbb{R}^n$, and $y \in \{1, ..., c\}$, along with a classifier $f : \mathbb{R}^n \to \{1, \cdots, c\}$, our goal is to assess the attribution methods $e : x_i \mapsto v_i$, where v_i denotes the feature obtained from each sample x_i. Now, we choose the pixel with the highest k% value from v_i and set the corresponding pixels in x_i to a fixed value of 0. This process can be represented as $\tilde{x}_i = (1 - M_i) \odot x_i = \pi_i(x_i)$, where M_i is the binary mask where erasing pixels are set to 1, and $\odot$ denotes element-wise multiplication. Then, we train the model on the modified dataset $\tilde{\mathcal{D}} = \{(\tilde{x}_i, y_i)\}_{i=1}^n$ and measure how much the accuracy decreases.

Remove-and-Debias (ROAD) [25]. After the introduction of ROAR, several studies identified potential pitfalls related to fixed value imputation. When the model is retrained on the modified dataset, it learns to rely on the mask-related information in $\tilde{x}_i$ to predict the class label. This makes the model exploit the spurious correlation between the mask and the label, preventing us from measuring how much information has been lost from x_i due to pixel perturbation.

Rong et al. [25] employ information theory and demonstrate that the problem stems from high mutual information between the modified image $\tilde{x}_i$ and the imputation mask M_i. They define this problem as *class information leakage*. Based on this analysis, ROAD adopts a different approach for pixel imputation, utilizing noisy linear interpolation instead of fixed values. With noisy linear

imputation, ROAD effectively removes the information of M_i from $\tilde{x}_i$, thereby preventing the model from predicting the class label based on mask-related information.

3 Geometric challenges in ROAR

In § 3.1, we analyze ROAR's evaluation mechanism from a geometric perspective. In § 3.2, we formalize the pitfalls of ROAR's pixel-perturbation strategy. In § 3.3, we attribute the cause of issues with ROAR to two main factors: the reliance on pixel-coordinate and the all-or-none removal processes.

3.1 Mechanism of ROAR

We formalize the mechanism of the accuracy drop in ROAR. For ease of discussion, we consider a dataset $\mathcal{D} = \{(x_1, 0), (x_2, 1)\}$ with only one data point for each of the binary labels, 0 or 1. Note that the input $x_i = (x_i^1, x_i^2, \cdots, x_i^n)$ is an n-dimensional vector, where the subscript denotes the sample index, and the superscript indicates vector elements, i.e., pixels. We assume the model mispredicts when the difference between perturbed $\tilde{x}_1$ and $\tilde{x}_2$ for all pixels is less than ϵ. This corresponds to a situation where $\tilde{x}_1$ and $\tilde{x}_2$ are too similar to distinguish.

Upon perturbing x_1 and x_2 with ROAR (replacing important pixels with zeros), let the difference between them be denoted as $dx = (dx^1, \ldots, dx^n)$. Among these dx^i values, we define coordinates with a magnitude greater than ϵ as *relevant*, while the rest as *irrelevant*. According to our assumption, irrelevant coordinates do not contribute to distinguishing between the two samples, as their differences are already smaller than ϵ. On the other hand, in the case of relevant coordinates, the information of only a single relevant coordinate is sufficient to make samples separable. Therefore, the accuracy drop in ROAR occurs when all relevant coordinates are erased for each sample.

3.2 Pitfalls of ROAR

From the findings of § 3.1, we uncover two main problems of ROAR. The first problem is that ROAR yields different results depending on the choice of coordinates. For example, we could represent the same dataset with the coordinate system such that $dx = (dx^1, 0, ..., 0)$. In this representation, if $v_1 = v_2 = (1, 0, \cdots, 0)$, then simply removing one pixel will result in performance degradation. However, if we choose coordinates that have two or more relevant pixels, there is no accuracy drop until all those pixels are removed. This example shows that even identical geometric situations can lead to differing ROAR results based on coordinate choices.

Pitfall 1

ROAR is *not* invariant to the coordinate transformation.

The second problem is that ROAR cannot distinguish differences in the feature vector at relevant coordinates. To illustrate this, we consider a dataset $\mathcal{D}$ defined by $dx = (1, 2, \epsilon)$, and assume two attribution methods, denoted as e and e'. Features derived from e are $v_1 = v_2 = (1, 2, 0)$, while those from e' are $v_1 = v_2 = (2, 1, 0)$. Ideally, one might anticipate ROAR to favor method e due to its feature alignment with the distinctive direction of the samples. However, the accuracy drop in ROAR only occurs after removing all the relevant coordinates, which in this case are the first two pixels. Therefore, both e and e' witness an accuracy drop after two pixels are removed, rendering them indistinguishable. This example highlights that variations in relevant coordinates do not influence ROAR's outcomes, making it unable to distinguish between methods e and e'. Importantly, this limitation indicates that the main purpose of ROAR, feature comparison, might remain unfulfilled.

Pitfall 2

ROAR is *not* discriminative to differences in features at relevant coordinates.

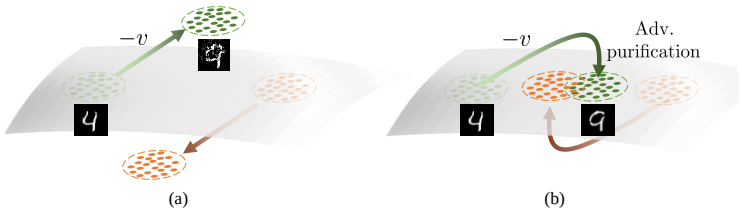

Figure 2: **Projection onto the data manifold**. Perturbation of input data (a) without manifold projection and (b) with manifold projection.

3.3 Origin of Pitfalls

Pitfall 1 arises because ROAR removes features from images based on pixel coordinates, ignoring the data's geometric structure. Therefore, to mitigate this issue, the pixel-perturbation strategy should be replaced with a *coordinate-free* perturbation approach.

The reason for Pitfall 2 is that when ROAR erases pixel information, it does so in an all-or-none manner. Because the information in a pixel remains intact until it is removed, it is possible to prevent an accuracy drop by retaining only those pixels that are perfectly intact. To resolve this issue, we need to *gradually* remove all features from the image.

4 Geometric Remove-And-Retrain (GOAR)

In this section, we introduce a novel evaluation metric using the feature-perturbation strategy: GOAR. In § 4.1, we explain that due to the presence of off-manifold components in the feature vector, naïve perturbation along the feature vector is not advisable. In § 4.2, we present how to remove off-manifold components after perturbation by utilizing the diffusion model. In § 4.3, we explain how to measure performance degradation in GOAR.

4.1 Feature perturbation

In § 3, our analysis reveals that the pitfalls of ROAR stem from its reliance on pixel coordinates and all-or-none manner removal. To overcome this drawback, we require a novel evaluation metric that perturbs images without depending on pixel coordinates.

A straightforward approach is shifting samples in the feature direction, denoted as $\tilde{x}_i = x_i - v_i$. However, as depicted in Figure 2, if the feature vector has an off-manifold component, this approach leads to samples escaping the data manifold [3, 37, 36]. When we retrain the model on this modified dataset, it tends to predict the class label based on the off-manifold feature in v_i rather than utilizing the remaining information in x_i. This problem can be viewed as a type of class information leakage that arises from significant mutual information between $\tilde{x}_i$ and v_i. To mitigate this problem, it is imperative to eliminate the off-manifold components.

4.2 Projection onto the data manifold

To eliminate the off-manifold component in $\tilde{x}_i = x_i - v_i$, projecting the perturbed image onto the data manifold is necessary. Our goal is to remove any off-manifold components in $\tilde{x}$ while preserving its on-manifold components. We approach the off-manifold component as an adversarial attack and apply an adversarial purification technique [21, 27, 42]. Especially, Nie et al. [21] found that adding minor noise to an image and subsequently denoising it via a diffusion model [10, 32, 34] maintains the image signal while efficiently counteracting the attack.

However, unlike conventional adversarial purification, which deals only with nearly imperceptible attacks, our method entails notable modifications to the image. Merely introducing minor noise fails to eliminate all off-manifold components. Yet, excessively large noise is discouraged as it not only purifies the image but also substantially modifies the data's significant components. Interestingly, we discovered that it is possible to perform additional adversarial purification by

treating v itself as noise. This can be understood as the diffusion model effectively captures and removes the off-manifold components mixed in v as a noise.

Specifically, our purification process is as follows: Starting with the perturbed image $x - v$, we introduce a small noise w, yielding $\tilde{x} = x - v + w$. Next, we employ the diffusion model to denoise $\tilde{x}$. This approach effectively eliminates irrelevant information associated with off-manifold directions while keeping the meaningful on-manifold components unchanged.

We present visual results in Figure A1. For details of the algorithm and hyper-parameter setting, please refer to Appendix B.

4.3 Performance degradation

ROAR and ROAD measure performance degradation by assessing how much a model's accuracy drops as perturbations are added. GOAR adopts a distinct measurement criterion, which involves tallying the cumulative number of misclassified examples. This approach is chosen because, in GOAR, a sample can regain separability after losing its information, effectively becoming indistinguishable from other class labels. Therefore, even after the image becomes distinguishable again, such cases are considered successful instances of performance degradation.

5 Experiments

In this section, we verify our analysis and highlight the effectiveness of GOAR with experiments. First, we use a toy dataset to demonstrate the shortcomings of the pixel-perturbation strategies for evaluating features, while showing that GOAR provides an accurate comparison (§ 5.1). Furthermore, we show that pixel-perturbation strategies have drawbacks in comparing various attribution methods in real datasets, whereas GOAR can appropriately compare these methods without such problems (§ 5.2).

In § 5.1, we conduct experiments on a synthetic Gaussian mixture dataset and use a 3-layer MLP model. In § 5.2, we conduct experiments on three image classification datasets: MNIST [6], FashionMNIST [40], and CIFAR10 [15]. We compare four attribution methods: Input gradient (Grad) [30], Grad $\times$ Input (I$\times$G) [29], SmoothGrad (SG) [31], and Integrated gradient (IG) [38]. As a control, we opt for a randomly generated vector (Random) as a feature, serving as a lower bound for performance comparison. We use a simple 3-layer CNN model for all experiments. For implementation details, please refer to Appendix B.

5.1 Feature assessment with a toy dataset

To test whether benchmarks can effectively compare the quality of features, we first create a 64-dimensional Gaussian mixture dataset, i.e., $x \sim \mathcal{N}(x; \mu_k, 0.3\mathbf{I})$. Here, data x, sampled from class 1 and 2, follow $\mu_1 = (1, ..., 1)$ and $\mu_2 = (-1, ..., -1)$, respectively. When trained on a simple neural network, the input gradients align parallel with the difference of the μ_k vectors for each class k. In other words, the input gradients effectively capture the ground truth features.

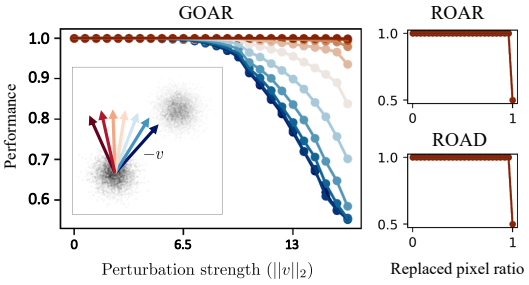

Figure 3: **Feature assessment with a toy dataset.** Comparison of features using ROAR, ROAD, and GOAR. The different colors correspond to the extent of disruption applied to the original feature, as shown in the inset of GOAR.

We then add noise of varying ratios to the features. As shown in Figure 3, as the noise ratio increases, the feature quality deteriorates. However, benchmarks like ROAR and ROAD show no difference in performance regardless of noise size. In essence, they are unable to differentiate between feature qualities. In contrast, we observe GOAR's performance rapidly deteriorating as the noise level increases, demonstrating its ability to accurately assess feature quality.

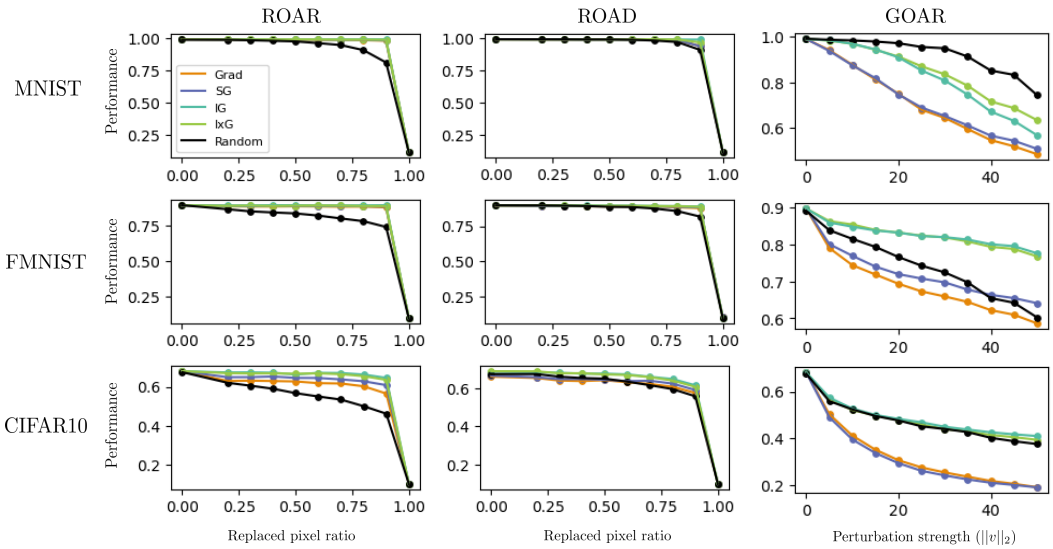

Figure 4: **Feature assessment with real datasets.** Comparison of attribution methods using ROAR, ROAD, and GOAR. Here, real image datasets of MNIST, FashionMNIST (FMNIST), and CIFAR10 are used for evaluating five attribution methods.

5.2 Feature assessment with real datasets

In Figure 4, we present the results of evaluating various attribution methods using ROAR, ROAD, and GOAR. ROAR and ROAD exhibit minimal differences in performance within attribution methods, making it nearly impossible to compare different methods, especially on the MNIST and FashionMNIST datasets. Furthermore, the fact that all methods perform even worse than the Random, which is a baseline, raises doubts about the results obtained through these approaches.

However, GOAR effectively allows the comparison of different methods. Our approach indicates that Grad and SG produce better feature vectors than methods involving input multiplication, such as IG and I$\times$G, while Grad and SG show almost no difference.

6 Discussions

GOAR overcomes the constraints of previous evaluation metrics like ROAR or ROAD, but it still has limitations stemming from its reliance on the diffusion model. First, employing the diffusion model is time-consuming. For specific computational costs, Appendix B can be referred. Additionally, when features have low-frequency components in their off-manifold direction, the diffusion model may struggle to effectively project the modified image onto the data manifold [21, 45]. Therefore, future research should focus on debiasing the off-manifold direction without relying on the diffusion model.

7 Conclusion

In conclusion, we analyzed pixel-perturbation metrics like ROAR and ROAD from a geometric perspective, revealing their limitations in comparing feature attribution methods. Their approaches are coordinate-dependent and fail to discriminate differences in features at relevant coordinates. To address these issues, we introduce a novel feature-perturbation metric named GOAR. With experiments on both toy and real datasets, we show that GOAR outperforms pixel-based approaches in comparing features. We hope our geometric analysis and metric help cultivate a new insight into feature attribution methods and their evaluation metrics.

References

[1] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations, 2020.

[2] Umang Bhatt, Adrian Weller, and José MF Moura. Evaluating and aggregating feature-based model explanations. *arXiv preprint arXiv:2005.00631*, 2020.

[3] Sebastian Bordt, Uddeshya Upadhyay, Zeynep Akata, and Ulrike von Luxburg. The manifold hypothesis for gradient-based explanations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshop*, pages 3696–3701, 2023.

[4] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687*, 2022.

[5] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. *arXiv preprint arXiv:2206.00941*, 2022.

[6] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[7] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*, 2019.

[8] Daniel C Elton. Self-explaining ai as an alternative to interpretable ai. In *Artificial General Intelligence: 13th International Conference, AGI 2020, St. Petersburg, Russia, September 16–19, 2020, Proceedings 13*, pages 95–106. Springer, 2020.

[9] Peter Hase and Mohit Bansal. Evaluating explainable ai: Which algorithmic explanations help users predict model behavior?, 2020.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[11] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks, 2019.

[12] Aya Abdelsalam Ismail, Mohamed Gunady, Hector Corrada Bravo, and Soheil Feizi. Benchmarking deep learning interpretability in time series predictions. *Advances in neural information processing systems*, 33:6441–6452, 2020.

[13] Jeya Vikranth Jeyakumar, Joseph Noor, Yu-Hsi Cheng, Luis Garcia, and Mani Srivastava. How can i explain this to you? an empirical study of deep neural network explanation methods. *Advances in Neural Information Processing Systems*, 33:4211–4222, 2020.

[14] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restor ation models, 2022.

[15] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. URL `https://api.semanticscholar.org/CorpusID:18268744`.

[16] Minjong Lee and Dongwoo Kim. Robust evaluation of diffusion-based adversarial purification. *arXiv preprint arXiv:2303.09051*, 2023.

[17] Yang Liu, Sujay Khandagale, Colin White, and Willie Neiswanger. Synthetic benchmarks for scientific research in explainable machine learning. In *Advances in Neural Information Processing Systems Datasets Track*, 2021.

[18] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[19] Pavan Rajkumar Magesh, Richard Delwin Myloth, and Rijo Jackson Tom. An explainable machine learning model for early detection of parkinson's disease using lime on datscan imagery. *Computers in Biology and Medicine*, 126:104041, 2020.

[20] Sara Mirzavand Borujeni, Leila Arras, Vignesh Srinivasan, and Wojciech Samek. Explainable sequence-to-sequence gru neural network for pollution forecasting. *Scientific Reports*, 13(1): 9940, 2023.

[21] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. Diffusion models for adversarial purification. In *International conference on machine learning*, 2022.

[22] 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-W*, 2017.

[23] An phi Nguyen and María Rodríguez Martínez. On quantitative aspects of model interpretability, 2020.

[24] Marco Tulio Ribeiro, Sameer Singh, and Carlos 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.

[25] Yao Rong, Tobias Leemann, Vadim Borisov, Gjergji Kasneci, and Enkelejda Kasneci. A consistent and efficient evaluation strategy for attribution methods, 2022.

[26] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[27] Changhao Shi, Chester Holtz, and Gal Mishne. Online adversarial purification based on self-supervised learning. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=_i3ASPp12WS`.

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

[29] Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. Not just a black box: Learning important features through propagating activation differences, 2017.

[30] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

[31] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise, 2017.

[32] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=St1giarCHLP`.

[33] Junhwa Song, Keumgang Cha, and Junghoon Seo. On pitfalls of remove-and-retrain: Data processing inequality perspective. *arXiv preprint arXiv:2304.13836*, 2023.

[34] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=PxTIG12RRHS`.

[35] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[36] Suraj Srinivas and Francois Fleuret. Rethinking the role of gradient-based attribution methods for model interpretability. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=dYeAHXnpWJ4`.

[37] Suraj Srinivas, Sebastian Bordt, and Hima Lakkaraju. Which models have perceptually-aligned gradients? an explanation via off-manifold robustness. *arXiv preprint arXiv:2305.19101*, 2023.

[38] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[39] Richard Tomsett, Dan Harborne, Supriyo Chakraborty, Prudhvi Gurram, and Alun Preece. Sanity checks for saliency metrics. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 6021–6029, 2020.

[40] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[41] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (in)fidelity and sensitivity for explanations, 2019.

[42] Jongmin Yoon, Sung Ju Hwang, and Juho Lee. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*, pages 12062–12072. PMLR, 2021.

[43] Ramy A Zeineldin, Mohamed E Karar, Ziad Elshaer, · Jan Coburger, Christian R Wirtz, Oliver Burgert, and Franziska Mathis-Ullrich. Explainability of deep neural networks for mri analysis of brain tumors. *International journal of computer assisted radiology and surgery*, 17(9):1673–1683, 2022.

[44] Wei Zhang, Ziming Huang, Yada Zhu, Guangnan Ye, Xiaodong Cui, and Fan Zhang. On sample based explanation methods for nlp: Efficiency, faithfulness, and semantic evaluation. *arXiv preprint arXiv:2106.04753*, 2021.

[45] Zhengyu Zhao, Hanwei Zhang, Renjue Li, Ronan Sicre, Laurent Amsaleg, and Michael Backes. Towards good practices in evaluating transfer adversarial attacks. *arXiv preprint arXiv:2211.09565*, 2022.

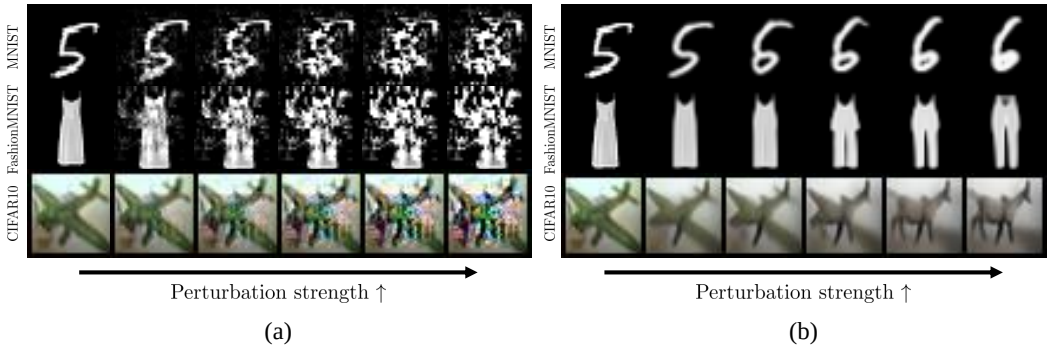

Figure A1: **Perturbed images on the data manifold.** Perturbed images of MNIST, FashionMNIST, and CIFAR10 (a) without manifold projection and (b) with manifold projection. Perturbation strength increases from left to right in the columns.

A Related Works

Feature attribution methods. Feature attribution methods aim to discover the relevant feature of given input that influences the model's prediction behind the scenes. These can be categorized into (i) removal-based approaches and (ii) gradient-based approaches. Removal-based approaches [18, 24, 28] involve removing parts of the input and extracting the features the model focuses on based on their impact. Gradient-based approaches [26, 35, 30, 31, 38, 29] find feature vectors that the model reacts to sensitively by using gradients with respect to the input or intermediate latent space. In this study, we use input gradient [30], SmoothGrad [31], Integrated gradient [38], and Grad $\times$ Input [29] for the comparison.

Evaluation Metric of the Feature Attribution Method Various evaluation benchmarks have been proposed so far, such as sanity checks [39, 1], human explanations [13], or synthetic datasets [17] with known ground truth features. ROAR [11] is one of the standard metrics that evaluates the feature maps extracted by XAI methods by removing a certain percentage (k%) of important pixels based on these feature maps and observing the resulting model accuracy drop on the modified dataset. When the same k% leads to a larger performance drop, the corresponding feature map is considered better.

Several works have highlighted concerns regarding ROAR and made efforts to resolve them. [25, 33, 2, 12, 7, 44] For instance, ROAD [25] highlights that when important pixels are masked with fixed values during removal, the information in the mask can be used in the prediction during the retraining phase, leading to label leakage. To address this, ROAD proposes noisy linear imputation. Many studies have critiqued ROAR from an information theory perspective. However, our paper is the first attempt to address the geometric issues in the ROAR process.

Manifold projection with diffusion model Various methods have been proposed to re-project images that have deviated from the image manifold using the diffusion model. One of the most prominent approaches is the field of inverse problems [14, 5, 4], which aims to recover the original source when the observed results are noisy. These methods suggest conditional image generation given observed images. Another area of research is adversarial purification [21, 16]. In a study by Nie et al. [21], it was discovered that adding small noise and then denoising with the diffusion model can effectively remove adversarial attacks. Our research adapts and modifies this second approach to suit our situation.

B Implementation details

Classifier For experiments with the real dataset, we employed a 3-layer CNN architecture. Each layer had channel numbers 32, 64, and 64, with kernel sizes of 5, 3, and 3, respectively. After the CNN layers, the output went through two fully connected (FC) layers. These FC layers had

widths of 64 and 32, respectively. The hyperparameters used for training were as follows: learning rate $= 3 \times 10^{-4}$, batch size $= 256$, Adam optimizer, and early stopping criteria $= 5$.

Diffusion model For all experiments, we used the DDPM [10] model. We use the DDIM [32] sampler for inference. The total number of inference steps is 25.

Adversarial purification To effectively remove the off-manifold component in v, we add additional noise w to perturbed image $x - v$. We add noise of a magnitude that increases the diffusion timestep by $0.16T$, where T is a maximum diffusion timestep, i.e. 1000. It corresponds to the additional DDIM inference step of 4.

Computational cost High computational cost is one of the significant limitations of GOAR. Retraining for one perturbation takes approximately minutes, while diffusion model sampling takes about minutes. Therefore, future work is needed to either avoid using the diffusion model or reduce the computational cost.

C Algorithm

In this section, for reproducibility, we provide the PyTorch [22] code for manifold projection. The source code for our experiments will be publicly available upon publication.

Manifold projection First, we determine the diffusion timestep t_{x-v} that matches the magnitude of the added v. Next, we add noise w of the predetermined size to x and adjust the timestep accordingly. In this case, the timestep becomes $t_{x-v+w} = t_{x-v} + 0.16T$. Finally, we sample $x - v + w$ with DDIM from t_{x-v+w} to 0.

```
import torch

def perturbation_strength_to_t(perturbation_strength, scheduler, in_ch=3, diffusion_timestep):
    '''
    Args
        perturbation_strength : l2 norm of v
        scheduler : huggingface diffusers scheduler
        in_ch : number of input channels
    '''
    noise_to_signal_ratio = (1-scheduler.alphas_cumprod).sqrt() / scheduler.alphas_cumprod.sqrt()
    dnsr = noise_to_signal_ratio - np.abs(remove_size) * self.init_trans.std.mean() / 0.5 / np.sqrt(in_ch*32*32)
    t    = dnsr.abs().argmin() / diffusion_timestep
    return t

def t_to_t_idx(t, scheduler, diffusion_steps=1000):
    return (scheduler.timesteps - diffusion_steps*t).abs().argmin()

def adv_purification(x, diffusion_model, scheduler, perturbation_strength, additional_noise_t_idx=4):
    '''
    Args
        x : perturbed image, i.e., x - v
        diffusion_model : pretrained diffusion model
        perturbation_strength : norm of v
        scheduler : DDIM scheduler, you could find it in Huggingface diffusers library
        additional_noise_t_idx : the size of additional noise for better purification
    '''
    # fix random seed
    generator = torch.Generator().manual_seed(SEED)

    # define timesteps
```

```
    scheduler.set_timesteps(num_inferences=25)

    # estimate proper diffusion timestep for x-v+w
    t_x = perturbation_strength_to_t(remove_size, scheduler)
    t_idx_x = t_to_t_idx(t_x, scheduler)
    t_idx_x_plus_noise = t_idx_x+additional_noise_t_idx
    t_x_plus_noise = scheduler.timesteps[t_idx_x_plus_noise]

    # add additional noise for better adversarial purification
    noise_signal_ratio = (1-scheduler.alphas_cumprod).sqrt() /
    scheduler.alphas_cumprod.sqrt()
    xt = x + (noise_signal_ratio[t_x_plus_noise] - noise_signal_ratio[
    t_x]) * torch.randn(x0.shape, generator=generator)
    xt = torch.sqrt(self.alphas_cumprod[t_x_plus_noise]) * xt

    # DDIM sampling (xt -> x0)
    for t in enumerate(scheduler.timesteps):
        if t > t_x_plus_noise:
            continue
        et = diffusion_model(xt, t)
        xt = scheduler.step(et, t, xt, eta=0, generator=generator).
    prev_sample
    x0 = xt

    return x0
```

Listing 1: **Adversarial purification**