HERMITRY RATIO: EVALUATING THE VALIDITY OF PERTURBATION METHODS FOR EXPLAINABLE DEEP LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Perturbation methods are model-agnostic methods used to generate heatmaps to explain black-box algorithms such as deep neural networks. Perturbation methods work by perturbing the input image. However, by perturbing parts of the input image we are changing the underlying structure of the image, potentially generating out-of-distribution (OOD) data. This would violate one of the core assumptions in supervised learning, namely that the train and test data come from the same distribution. In this study, we coin the term hermitry ratio to quantify the utility of perturbation methods by looking at the amount of OOD samples they produce. Using this metric, we observe the utility of XAI methods (Occlusion analysis, LIME, Anchor LIME, Kernel SHAP) for image classification models ResNet50, DensNet121 and MnasNet1.0 on three classes of the ImageNet dataset. Our results show that, to some extent, *all* four perturbation methods generate OOD data regardless of architecture or image class. Occlusion analysis primarily produces in-distribution perturbations while LIME produces mostly OOD perturbations.

1 INTRODUCTION

In recent years there has been an explosion of explanation methods for artificial intelligence (XAI) algorithms, largely due to the fact that AI is used in pretty much every facet of our modern lives. The need for trust and understanding in AI algorithms has never been more important. A popular way to create model explanations is to repeatedly remove features from the input and seeing how this affects the model prediction. Depending on the difference in prediction, this feature is assigned an importance score. The higher the score, the more the removal of this feature (or set of features) impacts the model prediction. These XAI methods are commonly referred to as perturbation or feature importance methods and they produce an explanation in the form of a heatmap or a saliency map (Zeiler and Fergus, 2014; Zhou et al., 2015; Ribeiro et al., 2016; 2018; Lundberg and Lee, 2017). These methods define *model explanation* as an importance weighting of all the features in the input data conditioned on the model prediction. For an overview of methods that will be analyzed see Table 1.

Table 1: Perturbation methods investigated in this paper.

Method	**Reference**
LIME	Ribeiro et al. (2016)
Anchor LIME	Ribeiro et al. (2018)
Kernel SHAP	Lundberg and Lee (2017)
Occlusion sensitivity	Zeiler and Fergus (2014); Zhou et al. (2015)

Perturbation methods are also used to evaluate the quality of heatmaps, regardless of whether or not the heatmap in question was obtained through a perturbation method (Hooker et al., 2019; Samek et al., 2017; Kindermans et al., 2018; Petsiuk et al., 2018). These evaluation methods remove areas from the input specified to be important by the heatmap. The perturbed input is fed to the model and the performance of the model is compared to the performance of the model when the input was not perturbed. The larger the discrepancy in performance, the more correct the heatmap is deemed to be.

There are various benefits to perturbation methods: easy to compute, model agnostic and intuitively interpretable to many users. However, as indicated in previous work (Hooker et al., 2019; Sundararajan et al., 2017), the perturbation of the input after training may well be presenting the trained model with data that is out-of-distribution (OOD). By perturbing a data sample, we introduce changes to its underlying structure that can be considered OOD. This violates one of the main assumptions when training machine learning (ML) models in a supervised manner: The training and evaluation data must come from the same distribution. If the model is fed input that is OOD, then we cannot make any conclusions nor give any explanations based on its output. If the perturbation methods are indeed generating OOD data, a reevaluation of their efficacy is required. In our paper we address the questions: **Are perturbation methods generating OOD data? If so, to what extent?**

To our knowledge, this is the first study that investigates whether perturbation methods generate OOD data. We focus our investigation to Deep Neural Networks (DNNs) trained with supervised learning on image classification tasks. Empirical results indicate that DNNs underperform, relative to their train and test performance, when they are presented with OOD data. This has spawned a whole field of OOD sample detection (Bulusu et al., 2020; Siegismund et al., 2020).

Contribution. In this paper we address the above stated questions by defining a new concept called *hermitry ratio*. We demonstrate that using this concept we can quantify and determine to what extent perturbation methods are generating OOD data when applied to DNNs trained on an image classification task using supervised learning. We test our method on four popular perturbation methods applied to three widely used neural architectures for three ImageNet data classes. Our results show that, to some extent, *all* four perturbation methods generate OOD data regardless of architecture or image class. This questions the general applicability of perturbation methods as well as the conclusions that can be drawn from them.

2 Related work

Kindermans et al. (2019) previously investigated the reliability of saliency methods and while these methods also produce in heatmaps, the methods investigated in the paper are not perturbation methods.

Previous work rarely addresses the specific issue in question. Hooker et al. (2019) propose an alternative to basic perturbation that involves re-training the model with a dataset that includes the perturbated data. While their results are good and their method is valid, in practice this method is very time consuming. Sundararajan et al. (2017) created the integrated gradients method keeping this in mind: "However, the images resulting from pixel perturbation could be unnatural, and it could be that the scores drop simply because the network has never seen anything like it in training."

3 Methods

3.1 Hermitry ratio

Analogous to the definition of a hermit, we use the term *hermitry*[1] to indicate how close a data sample is to a distribution. The larger the degree of hermitry, the further away from the group the data sample is. A data sample that is OOD is also called a hermit. This concept is illustrated in Figure 1.

Assume the training X and validation Y datasets where $x_i \in X, 0 < i \leq I_X$ and $y_i \in Y, 0 < i \leq I_Y$.

Let us assume a model M trained on X and its chosen encoding layer as L. We define the J dimensional encoding vector extracted from such a network as $h_{x_i} = M_L(x_i), h_{x_i} \in \mathbb{R}^J$.

To calculate hermitry, we rely on the distance of such an encoding to the training distributions encodings. Let us assume function $D_z = D(X, z), D_z \in \mathbb{R}$ describing the scalar distance between the distribution of encodings coming from the training dataset and the encodings of a sample z. Following that, we apply the same notion to a sample from the validation dataset to obtain $D_{y_i} = D(X, y_i)$ and to the whole dataset with $D_Y = D(X, Y), D_Y \in \mathbb{R}^{I_Y}$. Given that we have D_Y, let us assume a threshold T that defines the 95th percentile $n = \lfloor 0.95 I_Y \rfloor$ such that the sorted D_Y, $(D_Y)' = \{(D_Y)'_i | 1 < i \leq I_Y \wedge (D_Y)'_{i-1} \leq (D_Y)'_i\}$ satisfies $(D_Y)'_n \leq T$.

[1] To the best of our knowledge there is no alternative term that captures how much a data sample belongs to a distribution.

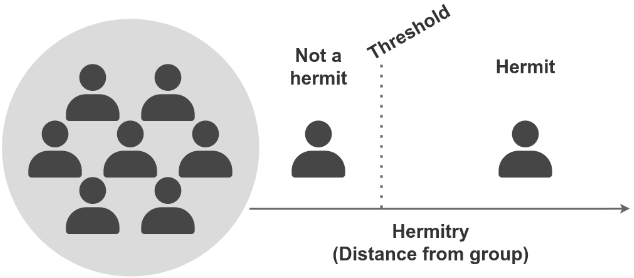


Figure 1: A hermit is someone who chooses to live alone, thus someone who is not a member of a group of other people. Analogous to this concept, we introduce the term hermitry to represent how close a data point is to a distribution. The further away a person (data point) is from a group (distribution), the more likely it is that they are a hermit (OOD). The threshold determines whether a data point is in-distribution (ID) or OOD. The value for this threshold is determined by the 95th percentile of validation class distance from the training class data.

Assuming a perturbation function ϕ and perturbed validation samples $\hat{Y} = \phi(Y)$, we define the distances of the perturbed validation samples as $\hat{D_Y} = D(X, \phi(Y))$ where $\hat{D_Y} \in \mathbb{R}^{\hat{I_Y}}$. Using the previously defined threshold T, we define the set of hermits as $\hat{D_Y^H} = \{\hat{D_Y} | \hat{D_Y} > T\}$ where $\hat{D_Y^H} \in \mathbb{R}^{\hat{I_Y^H}}$. Given the number of hermits, we define the *hermitry ratio* H as $H = \hat{I_Y^H} / \hat{I_Y}$.

In other words, we want to measure whether a perturbation method generates OOD data. Using hermitry we can measure how far one perturbed sample is from a dataset and determine if that sample is a hermit. However, we are not interested in only a single sample, but a collection of samples perturbed by the perturbation method. In this case it becomes more useful to look at the ratio of OOD perturbed data samples. The *hermitry ratio* H quantifies the ratio of perturbed samples that are OOD.

To determine what is a low H and what is a high H we empirically determined a cutoff value of 0.3. The reasoning for this is as follows: At $H = 0.5$ the XAI method generates OOD data for about half of the samples that it perturbs. This is an undesirable situation because about half of the perturbed samples are unreliable. By setting the value to 0.3 we allow for a number of samples to be OOD, which can happen because part of the original validation images are also considered to be OOD, see Appendix C. At 0.3 we say that 70% of the perturbed samples have to be ID. This cutoff value is used to analyze the results.

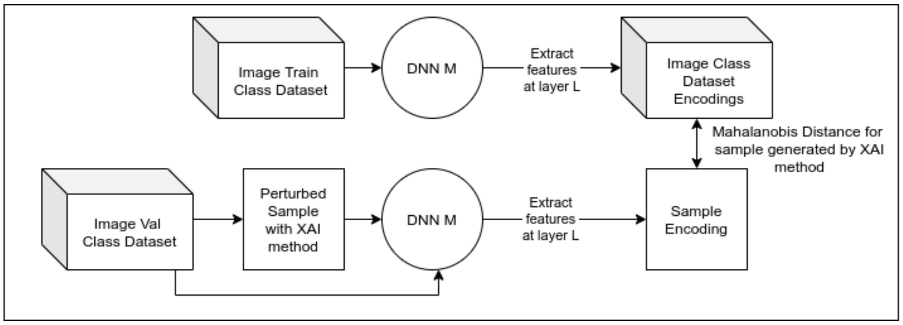


Image Train Class Dataset
DNN M
Extract features at layer L
Image Class Dataset Encodings
Mahalanobis Distance for sample generated by XAI method
Image Val Class Dataset
Perturbed Sample with XAI method
DNN M
Extract features at layer L
Sample Encoding

Figure 2: To measure the degree of hermitry of a data sample generated by a perturbation method, we measure the Mahalanobis distance from the encoding of the class dataset to the encoding of the perturbed sample. The larger the Mahalanobis distance, the further away from the true distribution the data is. The features from model layer L are used as encodings of the data samples.

3.2 Experiments

In total we run $4 \times 3 \times 3 = 36$ experiments to determine whether perturbation methods generate OOD data: 4 XAI methods, 3 models and 3 ImageNet classes. At the core of each experiment, we measure the distance between a perturbed sample encoding and the training class encodings. The perturbed encodings are based on the images in the validation class. This process is illustrated in Figure 2.

Distance metric. We use Mahalanobis distance to implement D (Lee et al., 2018; Çallı et al., 2019).

$$D(X, z) = \sqrt{(\mathbb{E}[h_X] - h_z)S^{-1}(\mathbb{E}[h_X] - h_z)}$$

where $h_X \in \mathbb{R}^{I_X \times J}$ and $\mathbb{E}[h_X] \in \mathbb{R}^J$ and $S = \text{cov}[h_X, h_X]$ where $S \in \mathbb{R}^{J \times J}$.

We stick to a class-conditional setting where the X and Y are chosen from samples that belong to the same class. Mahalanobis distance is not the only way to measure hermitry. ODIN (Liang et al., 2018) is another method for detecting OOD data samples. However, ODIN has the drawback that it perturbs the images, thus potentially also generating OOD data in the process.

Perturbation methods. We investigate four widely used perturbation methods listed in Table 1. Here we describe how these methods perturb the input image briefly. For more information and a discussion about these methods see Ras et al. (2020). For the first three methods we use the implementations provided by the Captum library (Kokhlikyan et al., 2020) and for Anchor LIME we use the open source implementation by Ribeiro et al. (2018). The parameters used to calculate the attributions using each of these methods are provided in Appendix A. *Occlusion sensitivity* sweeps a gray patch of fixed size across the image with the purpose of occluding pixel regions. In this paper we use a gray patch of 22 × 22 pixels. *Kernel SHAP* and *LIME* replace a subset of the image pixels with random pixels present in the data. *Anchor-LIME* samples regions in the image and replaces the pixels with a uniform color. Examples of perturbed images for each method can be seen in Table 2.

Models. To validate our method we wanted to see if our observations hold across various types of model architectures. We chose three different pre-trained models: ResNet50 (He et al., 2016), DenseNet121 (Huang et al., 2017) and MnasNet1.0 (Tan et al., 2019). Each architecture follows a different network building approach. We use model implementations from the PyTorch torchvision library (Paszke et al., 2019).

Encodings. To measure the distance between a single data point and a distribution of data points, we represent our data as encodings. We obtain these encodings by passing the data through a network and extracting the features at layer L, where layer L is the layer before the classification layer. This layer is chosen because images can contain multiple classes. Ideally, each image only contains the target class, however this is often not the case in ImageNet (Beyer et al., 2020). If the perturbation method hides information about the target class, the model tries to predict with the information available for the other class. This will be reflected in the classification layer as the shifting of the logit values towards the predicted class. It becomes impossible to determine if the encoding is OOD because the image was perturbed or because the model is predicting for a different class. Hence we go back to a layer that contains more information, which is layer L before the classification layer.

For the experiments we collect 8 types of encoding distances using Equation 3.2: Train-train, train-val, train-white noise, train-anime, train-LIME, train-Anchor LIME, train-Kernel SHAP, train-occlusion.

Train-train is used to validate that the dataset images are close each other. Train-val is used to determine the distance from the train encodings and the validation encodings. Train-val is important because we set the threshold shown in Figure 1 by calculating the value of the 95th percentile of train-val. Note that this value is unique per model and image class and it allows us to compare the perturbation methods with each other. In Figure 3 train-LIME, train-Anchor LIME, train-Kernel SHAP and train-occlusion are plotted as kernel density estimation (KDE) plots.

We intentionally feed the network what is usually considered OOD data and measure their distance to the train encodings. Train-white noise are the distances between the train encodings and encodings of uniform white noise images. Train-anime are the distances between the train encodings and encodings of a subset of the SFW Danbooru2020 Anime dataset (Anonymous et al., 2021). The resulting KDE plots can be found in Appendix C.

Dataset. In the experiments in this paper we use a class-conditional distance as described by Lee et al. (2018) to measure hermitry as opposed to distance to the entire dataset. I.e., Given class A, class B can be considered OOD. ImageNet images often contain multiple classes (Beyer et al., 2020). If an XAI method removes the information for one class, another class may become more likely. Thus we define OOD as not belonging to a specific class and use a class-conditional setting. We use the

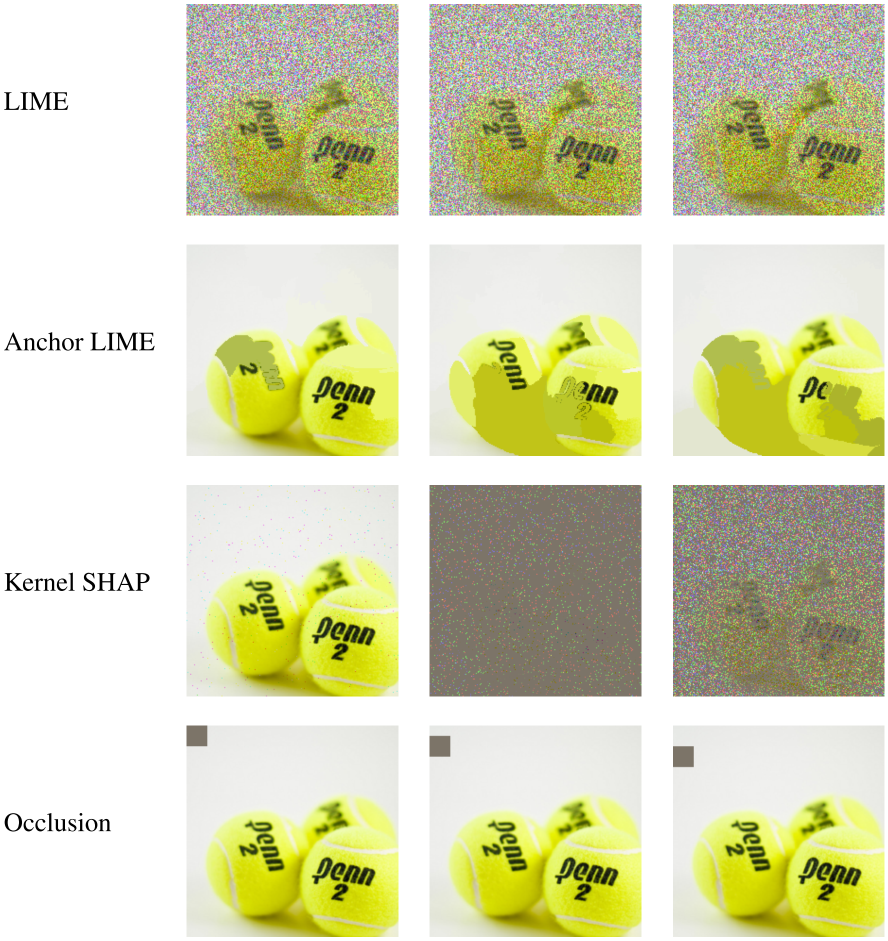


Table 2: Examples of what the perturbed images for each XAI method looks like, applied on the class Tennis Ball.

ImageNet dataset's official validation set to generate the perturbed samples. We run experiments on three arbitrarily chosen image classes: Tennis Ball, Printer and Chocolate Sauce. Examples from the validation dataset are provided in Figure 3.

4 Results

The hermitry threshold obtained by using the 95th percentile of the non-perturbed validation distances are provided in Appendix B. In Table 4 the hermitry ratio is listed for each perturbation method, model and ImageNet class. The table is divided into two parts, the top part shows the experiments resulting in a low hermitry ratio (≤ 0.3) while the bottom part shows the experiments resulting in a high hermitry ratio (> 0.3).

It is immediately clear that, to some extent, every chosen XAI method generates data that can be considered OOD. In every case tested the occlusion method generated data with a low hermitry ratio (mean hermitry ratio of 0.061), indicating that most of the time, the method does not produce data that can be considered OOD. The specific distribution of Mahalanobis distances relative to the training data class can be seen in Figure 3. Except for one case, LIME generates samples with a high hermitry ratio (mean hermitry ratio of 0.642). LIME generated samples with a low hermitry ratio when it was applied to MnasNet1.0 on the Tennis Ball class. Kernel SHAP (mean hermitry ratio of 0.315) and Anchor LIME (mean hermitry ratio of 0.325) produced mixed results. Kernel SHAP only has a low hermitry ratio when applied to MnasNet1.0 and Anchor LIME has a low hermitry ratio when

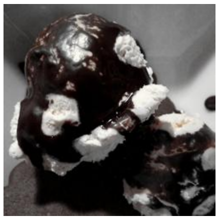
Chocolate Sauce			
Printer			
Tennis Ball			

Table 3: Example images from the classes that were used in this study.

applied to the Chocolate Sauce class. The results indicate that both the model architecture and the class influence whether an XAI method produces OOD data. Overall, when applied to MnasNet1.0 XAI methods produce low hermitry ratios (mean hermitry ratio of 0.195) compared to DenseNet121 (mean hermitry ratio of 0.387) and ResNet50 (mean hermitry ratio of 0.426) .

5 Discussion

In this study, we investigated four XAI perturbation methods to see if their perturbations are generating OOD data. The term hermitry is defined to describe how close to a distribution a data sample is. Using hermitry and the hermitry ratio we are able to quantify whether perturbation methods generate OOD data. First, we used images from the ImageNet training dataset to describe the expected encoding distributions for three classes. Using a subset of the training dataset, we described the expected encoding distributions for this class for a pre-trained deep learning model. Second, the distance between the validation samples and the encoded distribution was measured for each validation class, each class containing 50 samples. Third, we applied the four XAI methods to these 50 validation samples and generated encodings based on many perturbed image variations. Four, we compared the distances of the non-perturbed validation encodings to the perturbed ones to define the hermitry of the perturbed samples. We used the 95th percentile of the non-perturbed validation samples as a threshold for the hermitry of perturbed samples. By looking at the ratio of perturbed encodings after this threshold, we defined the hermitry ratio. We used this ratio to quantify the validity of the explanations made by an XAI method in a given situation. Finally, we repeat this process for three popular models and three arbitrarily chosen ImageNet classes.

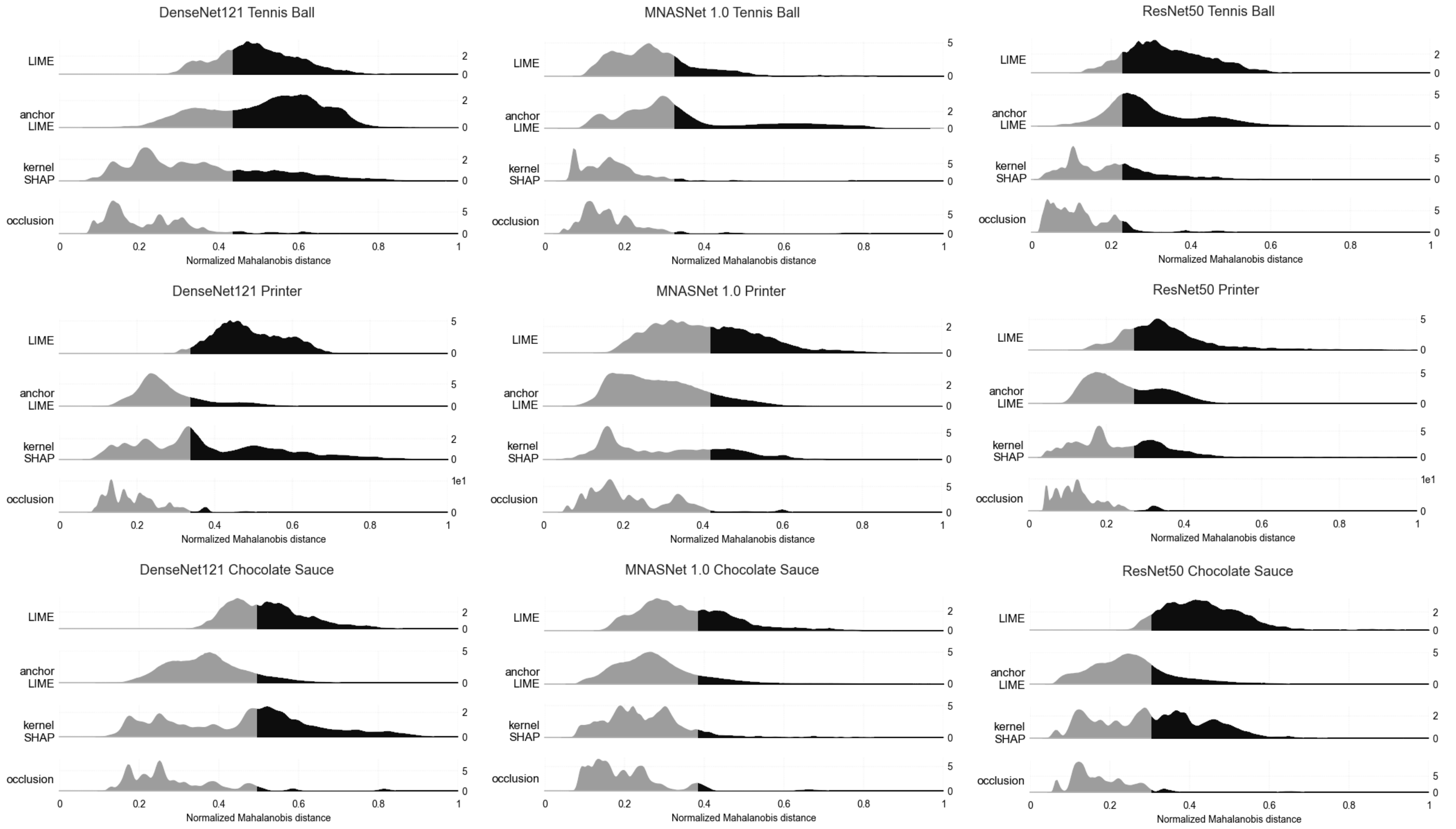


Figure 3: Mahalanobis distances (normalized) of perturbed sample encodings to the training dataset encodings. The gray parts of the distribution represent the samples that are in distribution with low hermitry. The black parts represent the samples that are OOD with high hermitry. The threshold is determined using the 95th percentile of the validation dataset's Mahalanobis distances.

Table 4: Hermitry ratio for each XAI method, model and class.

	Hermitry Ratio	XAI Method	Model	Class Name
Low Hermitry Ratio (≤ 0.3)	0.052	Occlusion	DenseNet121	Chocolate Sauce
	0.058	Occlusion	DenseNet121	Printer
	0.068	Occlusion	DenseNet121	Tennis Ball
	0.053	Occlusion	MnasNet1.0	Chocolate Sauce
	0.046	Occlusion	MnasNet1.0	Printer
	0.060	Occlusion	MnasNet1.0	Tennis Ball
	0.059	Occlusion	ResNet50	Chocolate Sauce
	0.060	Occlusion	ResNet50	Printer
	0.095	Occlusion	ResNet50	Tennis Ball
	0.260	LIME	MnasNet1.0	Tennis Ball
	0.084	Kernel SHAP	MnasNet1.0	Chocolate Sauce
	0.280	Kernel SHAP	MnasNet1.0	Printer
	0.062	Kernel SHAP	MnasNet1.0	Tennis Ball
	0.096	Anchor LIME	DenseNet121	Chocolate Sauce
	0.191	Anchor LIME	DenseNet121	Printer
	0.147	Anchor LIME	MnasNet1.0	Chocolate Sauce
	0.135	Anchor LIME	MnasNet1.0	Printer
	0.223	Anchor LIME	ResNet50	Chocolate Sauce
High Hermitry Ratio (> 0.3)	0.525	LIME	DenseNet121	Chocolate Sauce
	0.962	LIME	DenseNet121	Printer
	0.747	LIME	DenseNet121	Tennis Ball
	0.359	LIME	MnasNet1.0	Chocolate Sauce
	0.444	LIME	MnasNet1.0	Printer
	0.905	LIME	ResNet50	Chocolate Sauce
	0.722	LIME	ResNet50	Printer
	0.857	LIME	ResNet50	Tennis Ball
	0.447	Kernel SHAP	DenseNet121	Chocolate Sauce
	0.482	Kernel SHAP	DenseNet121	Printer
	0.311	Kernel SHAP	DenseNet121	Tennis Ball
	0.486	Kernel SHAP	ResNet50	Chocolate Sauce
	0.363	Kernel SHAP	ResNet50	Printer
	0.324	Kernel SHAP	ResNet50	Tennis Ball
	0.711	Anchor LIME	DenseNet121	Tennis Ball
	0.409	Anchor LIME	MnasNet1.0	Tennis Ball
	0.326	Anchor LIME	ResNet50	Printer
	0.688	Anchor LIME	ResNet50	Tennis Ball

It was unsurprising that occlusions produce data that is least OOD because occlusion was implemented with a patch of 22×22 pixels. We expect that increasing the patch size will also increase the degree of hermitry. Considering that LIME produces perturbations with a high level of noise it is also not surprising that, out of all the methods, it generated the most OOD data. Kernel SHAP produced a lower number of OOD data than LIME, even though some of the perturbed images look worse compared to LIME. We suspect that the number of images that are heavily perturbed make up only a small minority of the perturbed images. More investigation into this is needed. We expected the results for Anchor LIME to be similar to LIME, however, Anchor LIME produces fewer OOD data compared to LIME. One possible explanation is that the exploration approach guiding the perturbation is much more efficient in Anchor LIME compared to unguided random perturbations in LIME, leading to smaller, targeted perturbation regions.

Throughout the experiments it became clear that both the image class and the architecture choice affect to what degree the perturbation methods generate OOD data. E.g., 9 out of 12 MnasNet1.0 experiments produced low hermitry ratio, compared to the others 5 out of 12 for DenseNet121 and 4 out of 12 for ResNet50. One possible explanation is that MnasNet feature extraction is more robust to input perturbation, causing less changes to the encoding later in the network. For the moment it is unclear which architecture mechanism causes this phenomenon.

As for classes, there is currently not enough data to be able say anything about how the class choice is affecting the degree of hermitry.

Mentioned previously in Section 3.2 hermitry is defined and utilized in a class-conditional approach. Ideally we would like to not have it be class-conditional but rather encompass the entire dataset. For this to happen we need to improve the representation for the dataset encodings such that they correlate less with the network prediction, e.g., by extracting features from an earlier layer. Another approach is using a different image dataset where the images only contain a single class.

To quantify hermitry, we chose a hermitry threshold at the 95th percentile of the validation dataset distances. Considering that the perturbed samples also come from the validation dataset, this percentile makes sense. However, one may argue that this threshold point should be 99 or 90th percentiles. Although it is not perfect, 95th gives us a point to make comparisons on and the results would not have changed a lot when different thresholds were used.

Currently the cutoff for the hermitry ratio is at 0.3. Finding a more principled way of determining this cutoff could improve the validity of our findings.

In future work, we would like to produce a higher quality quantification of hermitry by extending our Mahalanobis distance application to reflect on the model predictions as per Lee et al. (2018), to distinguish OOD and different class. There are other input perturbation methods such as SHAPley Sampling Values (Strumbelj and Kononenko, 2010), feature permutation (Fisher et al., 2019), meaningful perturbation (Fong and Vedaldi, 2017), prediction difference analysis (Zintgraf et al., 2017), and representation erasure Li et al. (2016) that we can add to our analysis. Also there are many other model architectures that we can add to our analysis.

In general our study shows that perturbation methods that perturb a relatively small region of the image tend to generate less OOD data, while methods that perturb large regions tend to generate more OOD data. Quantifying the boundary for this amount of perturbation merits further investigation, as all perturbation methods can be improved if fitted with a boundary parameter that prevents excessive perturbation.

This study focuses on image classification, and our findings and conclusions are only applicable in this context. For other modalities such as text, or for other image datasets such as medical images, the conclusions may differ. Our results indicate that the underlying properties of the class and the model has an affect on the quantified hermitry ratio, hence utility of an XAI method. Identification of these properties and investigation of how they affect the hermitry ratio is crucial for understanding their utility for XAI methods.

REFERENCES

Anonymous, community, D., and Branwen, G. (2021). Danbooru2020: A large-scale crowdsourced and tagged anime illustration dataset. https://www.gwern.net/Danbooru2020. Accessed: 09/28/2020.

Beyer, L., Hénaff, O. J., Kolesnikov, A., Zhai, X., and Oord, A. v. d. (2020). Are we done with imagenet? *arXiv preprint arXiv:2006.07159*.

Bulusu, S., Kailkhura, B., Li, B., Varshney, P., and Song, D. (2020). Anomalous instance detection in deep learning: A survey. *arXiv preprint arXiv:2003.06979*.

Çallı, E., Murphy, K., Sogancioglu, E., and van Ginneken, B. (2019). Frodo: Free rejection of out-of-distribution samples: application to chest x-ray analysis. In *International Conference on Medical Imaging with Deep Learning–Extended Abstract Track*.

Fisher, A., Rudin, C., and Dominici, F. (2019). All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. *Journal of Machine Learning Research*, 20:1–81.

Fong, R. C. and Vedaldi, A. (2017). Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3429–3437.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Identity Mappings in Deep Residual Networks. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 630–645, Cham. Springer International Publishing.

Hooker, S., Erhan, D., Kindermans, P.-J., and Kim, B. (2019). A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., Erhan, D., and Kim, B. (2019). The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 267–280. Springer.

Kindermans, P.-J., Schütt, K., Alber, M., Müller, K.-R., Erhan, D., Kim, B., and Dähne, S. (2018). Learning how to explain neural networks: Patternnet and Patternattribution. In *International Conference on Learning Representations*.

Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., Melnikov, A., Kliushkina, N., Araya, C., Yan, S., et al. (2020). Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

Lee, K., Lee, K., Lee, H., and Shin, J. (2018). A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in Neural Information Processing Systems*.

Li, J., Monroe, W., and Jurafsky, D. (2016). Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Liang, S., Li, Y., and Srikant, R. (2018). Enhancing the reliability of out-of-distribution image detection in neural networks. In *International Conference on Learning Representations*.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*.

Petsiuk, V., Das, A., and Saenko, K. (2018). Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*.

Ras, G., Xie, N., van Gerven, M., and Doran, D. (2020). Explainable deep learning: A field guide for the uninitiated. *arXiv preprint arXiv:2004.14545*.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). 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. ACM.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *AAAI Conference on Artificial Intelligence*.

Samek, W., Binder, A., Montavon, G., Lapuschkin, S., and Müller, K.-R. (2017). Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*.

Siegismund, D., Heyse, S., and Steigele, S. (2020). Uncertainty with deep learning: a practical view on out of distribution detection. *Swiss Conference on Data Science (SDS)*, pages 65–66.

Strumbelj, E. and Kononenko, I. (2010). An efficient explanation of individual classifications using game theory. *The Journal of Machine Learning Research*, 11:1–18.

Sundararajan, M., Taly, A., and Yan, Q. (2017). Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, pages 3319–3328. JMLR. org.

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., and Le, Q. V. (2019). Mnasnet: Platform-aware neural architecture search for mobile. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2820–2828.

Zeiler, M. D. and Fergus, R. (2014). Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Zhou, B., Khosla, A., Lapedriza, À., Oliva, A., and Torralba, A. (2015). Object detectors emerge in deep scene cnns. In Bengio, Y. and LeCun, Y., editors, *International Conference on Learning Representations, Conference Track Proceedings*.

Zintgraf, L. M., Cohen, T. S., Adel, T., and Welling, M. (2017). Visualizing deep neural network decisions: Prediction difference analysis. In *International Conference on Learning Representations*.

APPENDIX A PARAMETERS FOR XAI METHODS

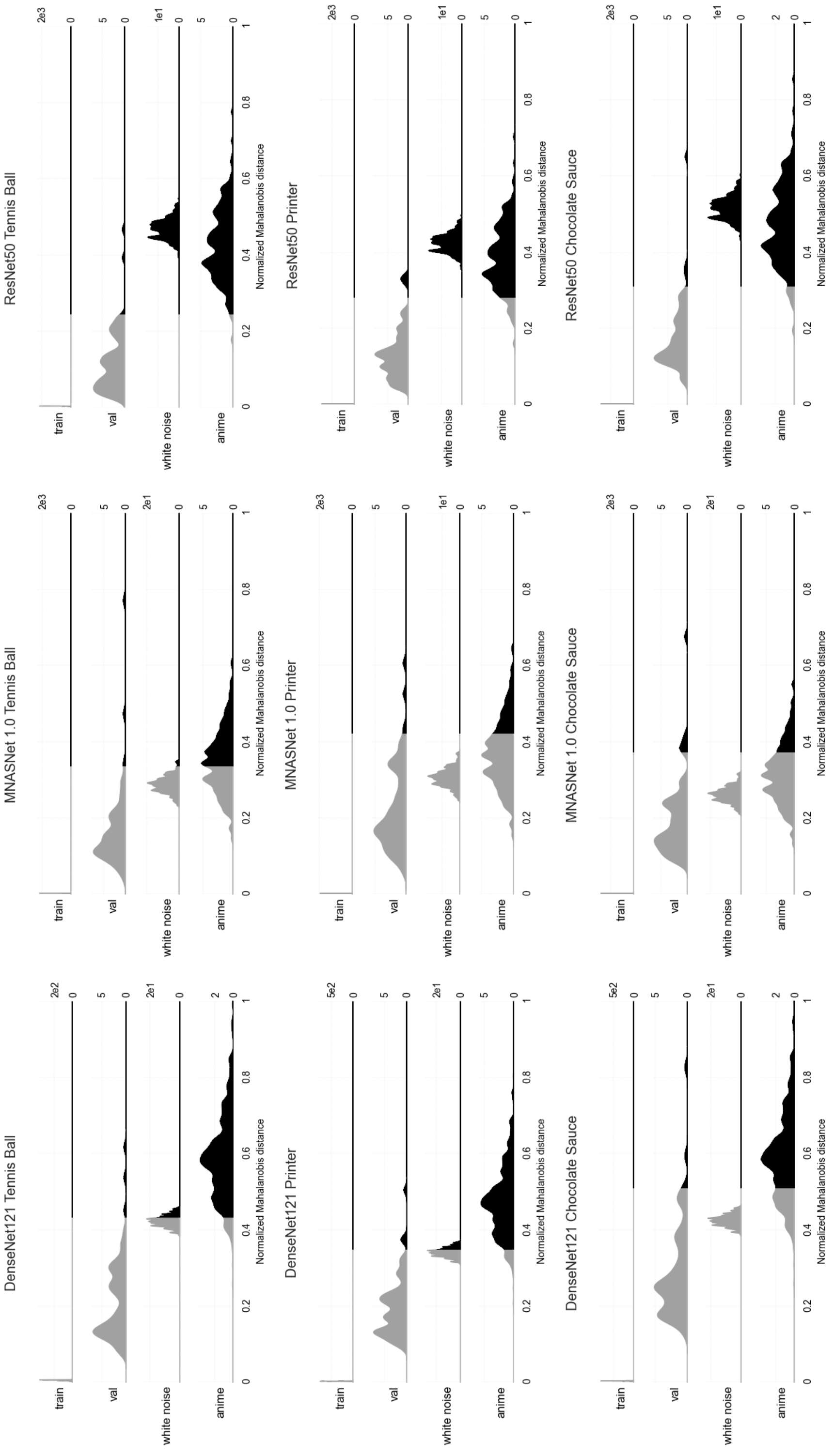
XAI Method / Parameter	Value
LIME	
interpretable_model	captum._utils.models.SkLearnLinearRegression()
similarity_func	captum.attr._core.lime.get_exp_kernel_similarity_function('euclidean', kernel_width=1000)
n_samples	100
Kernel SHAP	
n_samples	100
Occlusion	
sliding_window_shapes	(3, 22, 22)
strides	(3, 11, 11)
Anchor Lime	
library defaults	

APPENDIX B NORMALIZED HERMITRY THRESHOLDS

	Tennis Ball	**Printer**	**Chocolate Sauce**
DenseNet121	0.426262	0.351752	0.504520
ResNet50	0.234585	0.284813	0.315702
MnasNet1.0	0.322230	0.417694	0.386400

Appendix C Baseline density plots

Mahalanobis distances (normalized) of baseline sample encodings to the training dataset encodings. The gray parts of the distribution represent the samples that are in distribution with low hermitry. The black parts represent the samples that are OOD with high hermitry. The threshold is determined using the 95th percentile of the validation dataset's Mahalanobis distances.


ResNet50 Tennis Ball
ResNet50 Printer
ResNet50 Chocolate Sauce
MNASNet 1.0 Tennis Ball
MNASNet 1.0 Printer
MNASNet 1.0 Chocolate Sauce
DenseNet121 Tennis Ball
DenseNet121 Printer
DenseNet121 Chocolate Sauce
train
val
white noise
anime
Normalized Mahalanobis distance
2e3
5e2
2e2
1e1
2e1