[RE] Fooling Neural Network Interpretations via Adversarial Model Manipulation

Nikhil Krishna, Chenqing Hua, Yiran Wang
School of Computer Science, McGill University
[Nikhil.krishna, Chenqing.hua, Yiran.wang3]@mail.mcgill.ca

Abstract

In this work, we discover whether neural network interpreters can be fooled by manipulating adversarial models. We conduct rigorous and extensive experiments with certain interpretation methods, e.g. LRP, Grad-CAM, and discuss hyperparameters applied to models and foolings. Our results are validated by comparing the visual interpretations before and after the fooling and reporting quantitative metrics that measure the deviations from the original interpretations. The work shows that it is possible to manipulate adversarial models which can affect what models interpret regarding the cause of the prediction while not being noticed. We believe this work can facilitate developing a more robust and reliable neural network interpreter that can truly interpret the network's underlying decision-making process.

1 Introduction

Deep learning neural network models have achieved great results in many domains including computer vision. However, the neural network interpretation methods can be fooled by manipulating adversarial models. In this work, we conduct experiments with three pre-trained models implemented in the ImageNet competition, e.g. VGG19[1], ResNet50[2], and DenseNet121[3]. The adversarial model manipulation aims to fool the neural network interpretations without hurting the accuracy of the original models. By straightforwardly adding penalty terms to the interpreters, the interpretation results show that the two state-of-the-art interpreters can be fooled by adversarial model manipulation [4].

1.1 Neural network interpretation methods and adversarial model manipulation

Layer-wise Relevance Propagation (LRP)[5], Grad-CAM[6] are two well-known network interpretation methods that perform well on sanity checks among state-of-the-art interpretation algorithms. The two interpretation methods generate heatmaps to illustrate the importance of the input data to the prediction. The interpretations are fundamental to real-world applications which allow people to understand the relevancy of certain objects. In this work, we are interested in studying the reliability of the two interpretations by highlighting the true causes of the prediction. We propose *passive foolings* to study the reliability of the two interpretation methods. We show that it is possible to change the interpretations without hurting the prediction accuracy. That is, it is possible to have the right prediction but to lose important information. Training robust networks is an important problem that can lead to a better understanding of neural networks and improve their generalization capabilities.

1.2 Practical implication

The attack is practical since we manipulate the neural network models and intentionally prevent the foolings from being noticed by people. Such fooling methods are close to real-world applications. As a practical example, some medical applications are not only interested in the prediction but also understanding the cause of it. For example, we may seek to find out which parts of a medical image are most important for cancer diagnosis. In this case, correct interpretations are necessary for patients to have the right treatment. We believe our foolings can be generalized beyond object classification to empower an adversary to manipulate the reasoning about some medical diagnosis. For the reason, we can observe that fooled interpretations via adversarial model manipulations can cause some serious consequences regarding AI in real-world applications.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

1.3 Reproduction and modification

We reproduce a subset work of [4]. In our reproduction, we focus on the *passive* fooling in their work, as *active* fooling exceeded the computational capabilities of our systems. We implement three fooling methods to trick the two representative interpreters, LRP[5] and Grad-CAM[6]. We demonstrate that the interpreters are vulnerable when people intent to trick them. Moreover, after achieving the baselines, we consider possible changes to hyperparameters, including the learning rate and the penalty term. We propose a new set of hyperparameters that leads to a better quantitative metric score, in which the explanations of images (highlighted points) are changed more acutely via adversarial model manipulation. Moreover, we extend the experiments to find out the most vulnerable and the least reliable interpreter of the two interpretation methods. To summarize, we conduct a set of rigorous and extensive experiments to test the reliability and vulnerability of two representative interpretation methods.
Remark: In the original work [4], the original authors implemented three interpretation methods, LRP, G-CAM, and SimpleG. However, they mentioned in their paper [4] that the SimpleG$_T$ method generates very noisy heatmaps that particularly affect their observations. And they excluded the discussion of SimpleG$_T$ for manipulating the models. For this reason, SimpleG method is not explicitly discussed in our report. However, we will show that SimpleG$_T$ does generate noisy heatmaps.

2 Related Work

2.1 Interpretation methods of deep neural networks

Many interpretation methods for neural networks have been proposed, they are categorized as black-box methods and gradient/saliency map-based methods. In this related work, we introduce three gradient-based methods. The three methods are general approaches to explain predictions of neural networks. In 2013, Simonyan *et al.* first proposed Simple Gradient (SimpleG) in the paper "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps"[7]. Two years after, a new interpretation *Layer-Wise Relevance Propagation* (LRP) was presented in a work by Bach *et al.*[5]. The paper also showed that the SimpleG method usually generates noisier saliency maps. A year later, Selvaraju *et al.* introduced *Gradient-Weighted Class Activation Mapping* (Grad-CAM) in the paper "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization"[6].

2.2 Adversarial neural network

The adversarial idea of neural networks was introduced by Goodfellow *et al.*[8]. This framework corresponds to a minimax two-player self-play game. As a consequence, adversarial neural networks became one of the biggest topics in deep learning. Then, Szegedy *et al.* developed the adversarial neural networks to examine their stability and reliability[9]. The results showed that the state-of-the-art machine learning models can be fooled simply by applying easy back-propagation algorithms. Finally, Su *et al.* showed that modifying only one pixel using differential evolution is sufficient to fool classifiers[10]. Although there have been many proposed defense algorithms, people are smart enough to attack the models by modifying the attack algorithms.

In addition to building stable neural networks, people are also interested in finding reliable interpretation methods for neural networks. In 2018, Adebayo *et al.* proposed methodology to test sanity of saliency maps for neural networks[11]. Their work can be seen as an indispensable criterion for the interpreters. Finally, the paper we replicate proposed that people can fool the neural networks by intentionally changing their interpretations by manipulating adversarial models [4]. Their work can facilitate developing more robust and reliable neural networks and interpreters that benefit human beings.

3 Method and Hyperparameter

3.1 Background on Layer-wise Relevance Propagation (LRP) [5] and Grad-CAM [6]

The two interpretation methods are briefly discussed in this section. They generate heatmaps to highlight the regions of the image that have a significant effect in predicting the class.

LRP is an interpretation method that applies relevance propagation (similar to back-propagation). It generates heatmaps that show the relevance values of each pixel of the input image. The relevance values are positive or negative, indicating if a single pixel affects the prediction positively or negatively, respectively. In all the following experiments, a variant work of LRP, the LRP-Composite method is implemented. It applies the basic LRP-ϵ for the fully-connected layer and LRP-$\alpha\beta$ for the convolutional layer [12].
Grad-CAM is a CNN-based interpretation method that combines the gradient information and class activation maps to visualize the relevance of each data point to the prediction. Different from LRP, the important values of

Grad-CAM are only computed at the last convolutional layer, hence, the resolution of the visualization is much coarser than LRP [6].

3.2 Heatmap notations

We also denote the generated heatmaps. The training dataset for our supervised learning task is denoted as $D = \{(x_i, y_i)\}_{i=1}^{n}$, each input data is $x_i \in \mathbb{R}^d$ and each target label is $y_i \in \{1, ..., K\}$, where K is the number of classes. A neural network model is denoted as w. The generated heatmap from an interpretation method I for a neural network w and a class c is denoted as

$$h_c^I(w) = I(x, c; w) \tag{1}$$

in which $h_c^I(w) \in R^{d_i}$. If $d_I = d$, the j-th value of the heatmap, $h_{c,j}^I(w)$, represents the relevancy of input image x_i for the predicted class c.
Remark: For all our generated heatmaps, red and blue stand for positive and negative relevance values, respectively.

3.3 Objective function and penalty

Adversarial model manipulation is a method employed to fine-tune a model such that interpretation methods of the model are altered, but the accuracy of the predictions is not. The specific model manipulation method that is used utilizes cross-entropy classification loss along with a penalty term[4]. The penalty term is dependant on the interpretation method I that is used. The objective function we seek to minimize is a function of the training data D, the fooling dataset D_{fool}, the interpretation method I, the neural network w, and the parameters of the original pre-trained model w_0. Equation (2) shows the overall loss function.

$$\mathcal{L}(D, D_{fool}, I; w, w_0) = \mathcal{L}_C(D; w) + \lambda \mathcal{L}_{\mathcal{F}}^I(D_{fool}; w, w_0) \tag{2}$$

Note that in Equation (2), $\mathcal{L}_C(D; w)$ is the standard cross-entropy loss, $\mathcal{L}_{\mathcal{F}}^I(D_{fool}; w, w_0)$ is the penalty term specific to I, and λ is defined as a trade-off parameter (between normal loss and fooling penalty) [4]. For $\mathcal{L}_F^I(\cdot)$, the proposed passive fooling methods use different penalty terms outlined below.
Remark: In our work, $D_{fool} = D$; it is a simple notation for the dataset used in foolings.

3.3.1 Passive fooling

The fooling methods are proposed in the paper to trick neural network interpretations. We reproduce and extend the original work's passive fooling research. *Passive fooling* is defined as making a certain interpretation method output useless interpretations of predictions. The differences between the three passive fooling methods are simply in the penalty term described in the above section. The following subsections explain the interpretation method with its associated penalty term in the loss function $\mathcal{L}_{\mathcal{F}}^I$ [4].

Location fooling: This method aims to make explanations always emphasize certain areas of the input e.g. the corner of the image. The penalty term in (2) for this fooling method is defined as:

$$\mathcal{L}_{\mathcal{F}}^I(D_{fool}; w, w_0) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{d_I} ||h_{y_i}^I(w) - m||_2^2 \tag{3}$$

in which $D_{fool} = D$, $||\cdot||_2$ is the L_2 norm, $m \in R^{d_I}$ is a pre-defined mask vector that makes certain regions of the input important. Generally, we set m_i = 1 for the regions that we want to have a high importance, m_i = 0 for the areas to have a low importance.

Top-k fooling: This method aims to reduce the importance of the pixels that originally had the top k% highest values. The penalty term is defined as:

$$\mathcal{L}_{\mathcal{F}}^I(D_{fool}; w, w_0) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j \in \mathcal{P}_{i,k}(w_0)} |h_{y_i,j}^I(w)| \tag{4}$$

in which $D_{fool} = D$, $P_{i,k}(w_0)$ is the set of pixels of i-th input data point that had the top k% highest values for the original model w_0 in the explanations.

Center-mass fooling: The Center-mass method aims to deviate the center of mass of the heatmap as much as possible from the original one. The center of mass of a one-dimensional heatmap is denoted as $C(h_{y_i}^I(w)) = (\Sigma_{j=1}^{d_I} j \cdot h_{y_i,j}^I(w))/(\Sigma_{j=1}^{d_I} h_{y_i,j}^I(w))$, j is a location vector that can be easily extended to higher dimensions, $D_{fool} = D$, $||\cdot||_1$ is the L_1 norm, the penalty is defined as:

$$\mathcal{L}_{\mathcal{F}}^I(D_{fool}; w, w_0) = -\frac{1}{n} \sum_{i=1}^{n} ||C(h_{y_i}^I(w)) - C(h_{y_i}^I(w_0))||_1 \tag{5}$$

4 Experiment Setup

4.1 Dataset and models

The three main datasets we use in this work are D, D_{val} and D_{fool} as in the original work [4]. D is the ImageNet training data [13]. D_{val} is the entire ImageNet validation dataset for measuring the classification accuracy of the models and computing FSR. **Note**: In our experiments, $D_{fool} = D$, D_{fool} is just a notation for the fooling dataset. And we implement three pre-trained models, VGG[1], ResNet50[2], and DenseNet121[3] to carry out the foolings.
Remark: We generate the heatmaps of LRP, and Grad-CAM on a target layer, namely, the last convolution layer for VGG19, and the last block for ResNet50 and DenseNet121 as in paper[4]. We put the subscript T for LRP to denote such visualizations. The original authors[4] proposed that manipulating with LRP$_T$ was easier than with LRP and they have same effect.

4.2 Fooling Success Rate (FSR)

Fooling Success Rate is the special evaluation used in the following experiments to evaluate the performance of a fooling method f on an interpreter I [4]. It is a quantitative metric that indicates how much a fooling method f can affect an interpretation method I through model manipulation. For each fooling method, the test loss is introduced to evaluate FSR, it shows the gap between the current and target interpretations of each loss. Two parameters w_0 and w^*_{fool} are defined to indicate the pre-fooling test loss and the post-fooling test loss respectively; the interpreter I on each data point in the validation set D_{val}; the test loss for ith data point $(x_i, y_i) \in D_{val}$ is denoted as $t_i(w^*_{fool}, w_0, I)$ [4].

For *Location* and *Top-k* foolings, the losses for a single data point (x_i, y_i) and (w^*_{fool}, w_0) are evaluated by (3) and (4). And for Center-mass fooling, we evaluate (5) for a single data point (x_i, y_i) and (w^*_{fool}, w_0), and normalize it with the length of diagonal of the image to be the loss.

From the above losses, the FSR for a fooling method f applied to an interpretation method I is defined as:

$$FSR^I_{\mathcal{F}} = \frac{1}{|D_{val}|} \sum_{i \in D_{val}} 1\left\{t_i(w^*_{fool}, w_0, I) \in R_f\right\} \tag{6}$$

in which $1\{\cdot\}$ is an indicator function. R_f is a pre-defined interval for each fooling method. It is a threshold for determining whether a interpreter is successfully fooled or not. And R_f is defined to be $[0, 0.2]$, $[0, 0.3]$, and $[0.1, 1]$ for Location, Top-k, and Center-mass respectively, as in the paper [4]. A higher FSR metric indicates a more successful fooling method f for an interpreter I.

5 Passive Fooling Baseline Results (Reproduction)

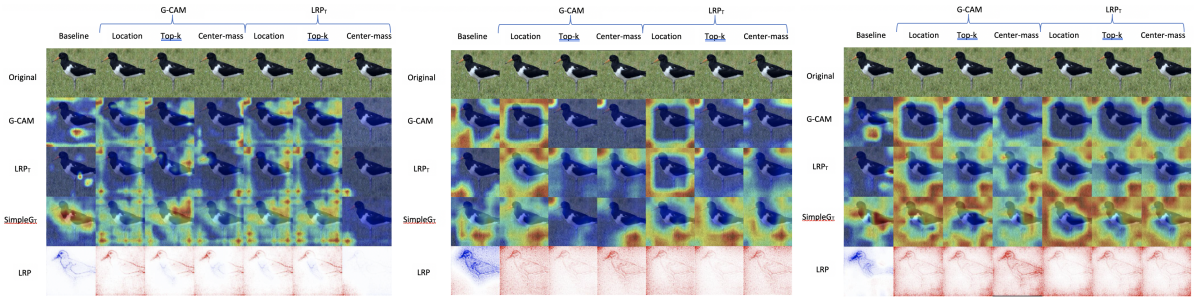
In Table (1), we show the hyperparameters that are used in our reproduction experiments (baselines). We reproduced the passive fooling section in the original paper [4], and have achieved the results of passive fooling as in their work.

Model	Hyperparameters	Location		Top-k		Center-mass	
		LRP$_T$	G-CAM	LRP$_T$	G-CAM	LRP$_T$	G-CAM
VGG19	lr	1e-6	1e-6	5e-7	3e-7	2e-6	1e-6
	λ	1	1	1	0.4	0.25	0.25
Resnet50	lr	1e-6	2e-6	4e-7	3e-7	6e-7	1e-7
	λ	4	2	1.5	4	0.5	1
DenseNet121	lr	2e-6	2e-6	1e-6	3e-6	5e-7	2e-6
	λ	2	2	1	6	0.25	0.25

Table 1: Hyperparameters of trained models for baseline experiments. lr stands for learning rate and λ is the regularization strength defined in (2).

In Figure (1), for location fooling, we observe that the highlighted pixels are moved to certain areas of the images after we emphasize the importance of those certain areas. The highlighted pixels are changed after the fooling, it shows the success of the fooling. For Top-k fooling, we see that the top k% most highlighted pixels are dramatically changed after the fooling by comparing the big difference between the original explanations and those in Figure (1). For center-mass fooling, the center of the heatmaps is changed to different parts of the images in Figure (1). The generated interpretations are mostly different from the original ones, and are incorrect.

Remark: In Figure (1), we generate heatmaps for SimpleG because we want to check whether SimpleG generates too noisy heatmaps that negatively affect our observations as mentioned in the original work [4]. This is indeed the case.



(a) VGG19 (b) ResNet50 (c) DenseNet121

Figure 1: Interpretations of the baseline and the passive fooled models for a 'Bird' image from the ImageNet validation set. The topmost row is the original image. Here, the baseline interpretations had no fooling done. Each first column for the three pre-trained models, VGG19 (1a), ResNet50 (1b) and DenseNet121 (1c), shows baseline interpretations by Grad-CAM, LRP$_T$, SimpleG$_T$ and LRP given the true class respectively. The next three columns are for the Grad-CAM interpreter that is used as I in the objective function (2) for three passive foolings i.e. Location, Top-k, and Center-mass fooling, respectively. The final three columns of each subimage are for the interpreter LRP$_T$ that is used as I in the objective function (2) for three passive foolings.

Model	FSR(%)	Location		Top-k		Center-mass	
		LRP$_T$	G-CAM	LRP$_T$	G-CAM	LRP$_T$	G-CAM
VGG19	G-CAM	0.8	**89.1**	31.5	**96.0**	49.9	**81.0**
	LRP$_T$	**87.4**	5.8	**96.2**	32.0	**99.8**	**66.3**
Resnet50	G-CAM	42.3	**97.3**	46.2	**99.9**	**66.0**	**67.3**
	LRP$_T$	**83.1**	0.9	**61.5**	5.0	**63.2**	0.8
DenseNet121	G-CAM	35.8	**81.8**	**62.5**	**98.3**	**66.8**	**72.4**
	LRP$_T$	27.0	0.4	**53.3**	1.7	**51.8**	21.8

Table 2: Fooling Success Rates (FSR) for passively fooled models. Underline is used for the FSRs of the matched interpreters used in foolings, and **Bold** indicates FSRs over 50%. For computing FSR, 10,000 images from ImageNet validation dataset are randomly selected. Since LRP and LRP$_T$ are variants, LRP is not included in the table; the FSR of LRP$_T$ illustrates whether LRP is fooled or not.

Model	Accuracy (%)	Pretrained	Location		Top-k		Center-mass	
			LRP$_T$	G-CAM	LRP$_T$	G-CAM	LRP$_T$	G-CAM
VGG19	Top1	72.4	71.8	71.4	71.6	72.3	70.5	70.6
	Top5	90.9	90.7	90.3	90.5	90.6	89.5	90.0
Resnet50	Top1	76.1	73.0	74.2	73.5	74.7	73.2	74.7
	Top5	92.9	91.3	91.8	91.7	92.0	91.7	92.1
DenseNet121	Top1	74.4	72.4	73.7	72.3	73.0	72.8	72.4
	Top5	92.0	91.0	91.7	91.0	90.9	91.0	91.0

Table 3: Accuracy of the pre-trained models and the manipulated models on the entire ImageNet validation set.

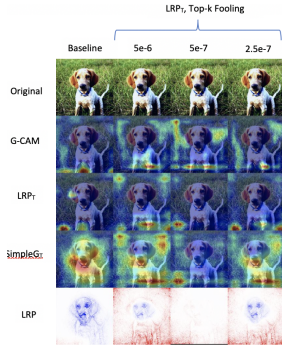
In Table (2), we can observe that all FSRs of fooling methods are more than 50% for any matched interpreters (bolded and underlined), except for the location fooling with LRP$_T$ for DenseNet121. The high FSR scores mean that we successfully fool the neural networks, as the models generate incorrect explanations. In Table (3), we can observe that the accuracy drops are around only 2% for Top-1 and 1% for Top-5 accuracy, the insignificant drops are difficult to observe. These results show that it is possible to manipulate adversarial models, which affects what these models interpret regarding the cause of the prediction. The FSR scores are evaluated over 10,000 images that are randomly sampled from the ImageNet validation dataset, it makes the results solid for their experiments. Passive foolings can mislead people, giving a wrong understanding of interpretations of neural networks because the most critical features are hidden and changed, and only less important features are highlighted.

6 Modification and Extension

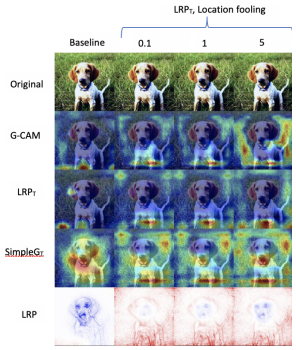
One important characteristic of strong research results is how flexible and robust they are in terms of changing certain hyperparameters. In this section, we modify and extend the original work [4] by testing hyperparameters and comparing the reliability of the interpretation methods.

6.0.1 Discussion of learning rate

This subsection discusses the effect of learning rate. For each fooling method for an interpreter on every model, we test three different learning rates, $10lr$, $1lr$, and $0.5lr$, where lr is the original learning rate of a given model in



(a) Examples of different lrs



(b) Examples of different λs

Figure 2: For Figure (2a), these are interpretations of the baseline and fooled VGG19 for a 'Dog' with three learning rates lr. The top row is the original picture. The first column is the baseline interpretations by Grad-CAM, LRP$_T$, SimpleG$_T$ and LRP given the true class respectively. The rest of the columns show the interpretation outputs for Top-k fooling with three learning rates, 5e-6, 5e-7 and 2.5e-7 (in order).
Figure (2b) shows the interpretations of the baseline and fooled VGG19 for a 'Dog' with three penalty values λ. The first column is the baseline interpretations by Grad-CAM, LRP$_T$, SimpleG$_T$ and LRP given the true class respectively. The rest of the columns show the interpretation outputs for Location fooling with three penalty values, 0.1, 1 and 5 respectively. For both subfigures, the LRP$_T$ interpreter was used.

Model	LRP$_T$	Top-k		
VGG19	lr	5e-6	5e-7	2.5e-7
	FSR(%)	97.0	96.2	96.0

Table 4: Top-k fooling applied to LRP$_T$ on VGG19 with three learning rates, and penalty value of $\lambda = 1$. 10,000 images from ImageNet validation dataset are randomly selected to compute FSRs.

baselines. We also fix the penalty value λ to 1. For example, the top-k fooling for LRP$_T$ on VGG19, the baseline learning rate lr is 5e-7, the three tested lr are 5e-6, 5e-7, and 2.5e-7, and λ is 1.

We compute FSRs for each fooling applied to every interpretation method on three models with different learning rates; we find that a greater learning rate can generally result in a higher FSR score. For instance, in Table (4), we can see that a greater learning rate results in a higher FSR. Also, by visually comparing images in Figure (2a), we see that the top k% highlighted pixels are more altered in the second column. The visual interpretations change more with a higher learning rate. We notice that the learning rate is strongly related to FSR. A higher learning rate can result in a more incorrect interpretation.

6.0.2 Discussion of penalty value

Model	LRP$_T$	Location		
VGG19	λ	0.1	1	5
	FSR(%)	75.0	87.4	91.1

Table 5: Location fooling applied to LRP$_T$ on VGG19 with three penalty values, and learning rate lr is 1e-6. 10,000 images from ImageNet validation dataset are randomly selected to compute FSRs.

We introduce the penalty term in (2). The penalty term is dependant on the interpretation method I. For every fooling method for an interpreter on every model, we discover three different penalty values, the values are 0.1λ, 1λ and 5λ respectively, where λ is the baseline penalty term in Table (1) for a given model. Additionally, the learning rates are fixed to 1e-6. For example, the Location fooling for LRP$_T$ on VGG19, the baseline penalty λ is 1, the three tested penalty values are 0.1, 1 and 5 respectively, and learning rate is 1e-6.

In Table (5), we can see that the FSR increases as the penalty value increases. After we compute and compare FSRs for each fooling method f applied to every interpreter I on three models with varied penalties, we find that greater penalty values can result in higher FSR scores. Also, in Figure (2b), we can observe that the highlighted points are more altered from important areas of the image to the corner areas; the visual explanations are more varied with a greater penalty value. As λ increases, the interpreters are fooled more and more; we conclude that the neural network interpreters can be more fooled with greater penalty values.

6.1 New set of hyperparameters and experiments

In general, a higher learning rate and a greater penalty value can result in better FSR scores. However, a too high learning rate and penalty value can hurt classification accuracy. It is important to find optimal hyperparameters

Model	Hyperparameters	Location		Top-k		Center-mass	
		LRP_T	G-CAM	LRP_T	G-CAM	LRP_T	G-CAM
VGG19	lr	1e-5	1e-5	5e-6	3e-7	1e-6	1e-5
	λ	5	5	5	2	1.25	1.25
Resnet50	lr	1e-5	1e-6	4e-6	3e-6	6e-6	1e-7
	λ	20	10	7.5	20	2.5	5
DenseNet121	lr	1e-6	1e-6	1e-5	3e-6	5e-6	1e-6
	λ	10	10	5	30	1.25	1.25

Table 6: Proposed new hyperparameters of trained models for extensions. The new set of learning rates lr and penalty values λ in our following experiments.

to maximize the goals. In Table (6), we propose new sets of hyperparameters after a thorough discovery. With implementation of our new sets of hyperparameters, the neural network interpreters are more fooled. However, the classification accuracy might have been slightly affected.

We randomly select 10,000 images from the ImageNet validation dataset to validate our proposed hyperparameters. From Table (7), we can observe that all FSRs of fooling methods for every matched interpreter (bold and underlined) have increased with our new hyperparameters compared to those in Table (2); most of remaining FSRs have also increased dramatically. The higher FSR scores mean that our proposed new hyperparameters can make the interpreters generate more incorrect explanations.

Model	FSR(%)	Location		Top-k		Center-mass	
		LRP_T	G-CAM	LRP_T	G-CAM	LRP_T	G-CAM
VGG19	G-CAM	42.5	**95.3**	35.7	**98.2**	**60.3**	**83.2**
	LRP_T	**92.9**	28.0	**96.3**	39.7	**99.9**	**78.8**
Resnet50	G-CAM	**52.2**	**98.2**	**52.4**	**99.9**	**70.8**	**73.5**
	LRP_T	**87.5**	15.9	**67.4**	23.0	**68.5**	20.2
DenseNet121	G-CAM	51.6	**90.8**	**65.3**	**99.0**	**70.3**	**75.6**
	LRP_T	40.6	9.8	**57.8**	17.8	**54.9**	30.8

Table 7: New Fooling Success Rates (FSR) for passively fooled models with proposed new hyperparameters. Underline is used for the FSRs of the matched interpreters used in foolings, and the **Bold** stands for the FSRs over 50%. 10,000 images from ImageNet validation dataset are randomly selected to compute FSRs.

Model	Accuracy (%)	Pretrained	Location		Top-k		Center-mass	
			LRP_T	G-CAM	LRP_T	G-CAM	LRP_T	G-CAM
VGG19	Top1	72.4	67.8	67.2	66.6	70.1	67.5	67.0
	Top5	90.9	89.6	89.4	90.4	89.6	88.5	90.2
Resnet50	Top1	76.1	68.0	68.1	69.5	70.4	69.0	71.0
	Top5	92.9	90.0	90.7	91.2	91.8	91.0	91.3
DenseNet121	Top1	74.4	70.3	69.0	70.1	68.8	69.5	69.0
	Top5	92.0	90.0	90.5	91.0	90.7	90.1	90.7

Table 8: Accuracy of the pre-trained models and the manipulated models with new hyperparameters on the entire ImageNet validation set. Notice that the accuracy drops are around 5% for Top-1 and 2% for Top-5 accuracy.

Moreover, we validate our proposed hyperparameters on the entire ImageNet validation dataset by comparing the accuracy. The significance of our proposed hyperparameters lies in the fact that the classification accuracy of all manipulated models are around the same as that of the pre-trained accuracy shown in Table (8). We can see that the accuracy drops are around 5% for Top-1 and 2% for Top-5 compared to pre-trained models'. Furthermore, we compare our accuracy to those in baseline experiments (see Table (3)), and notice decreases in accuracy. This is a trade-off between the FSR values and classification accuracy. If we want the interpretation methods to be more fooled, then it is likely to lower the classification accuracy. The FSRs in Table (7) show that our new hyperparameters can affect more of the models' interpretations regarding the cause of the prediction; however, the proposed hyperparameters also slightly hurt the accuracy as seen in Table (8).

6.2 Reliability and vulnerability of interpreters

6.2.1 Experiment

We extend three fooling methods to test the reliability and vulnerability of the two interpretation methods. The problem is important and practical because interpretation methods are close to real-world applications. People sometimes need the explanations and interpretations of neural networks. In this section, we design rigorous experiments to find the more stable and reliable interpretation method from the two interpreters that potentially help build real-world applications and better interpreters.

To compare the performance of two different interpretation methods, it is necessary to fix hyperparameters and the fooling method. We compute and compare FSR values for location fooling applied to each interpreter I on the three models VGG19[1], ResNet50[2] and DenseNet121[3] respectively with 1e-5 learning rate and $\lambda = 10$ penalty value over randomly sampled 10,000 images from ImageNet validation dataset. The choice of our experimental learning rate and penalty value results in higher FSRs, and can visually show clear differences in heatmaps and

demonstrate differences in FSRs for two interpretation methods. We set the target layer that generates heat maps to be the last convolution layer for VGG19 and the last block for both ResNet50 and DenseNet121.

6.2.2 Result

In Figure (3), we observe that the highlighted pixels are moved from important areas of the image to the corner areas after we emphasize the importance of corner areas by Location fooling. The generated interpretations are mostly different from the original ones, and are incorrect. We visually compare the heatmaps of G-CAM and LRP_T, and see that more highlighted pixels are moved to the corner areas in G-CAM. The heatmaps show that G-CAM interpreter is more likely to be fooled than LRP_T, especially looking at the middle row.

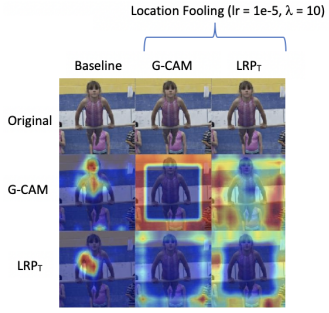


Figure 3: Interpretations of the baseline and the Location fooling by G-CAM and LRP_T for the model VGG19 for a 'Girl' image from the ImageNet validation set. The columns correspond to when no fooling was applied, when G-CAM was used, and when LRP_T was used, in order. For this experiment, $\lambda = 10$, lr =1e-5, and location fooling was used. When the interpreters match, note that the foolings become more apparent, but G-CAM still gets fooled more.

Furthermore, we compute FSRs to demonstrate and compare the performance of two interpreters. In Table (9), we can see that G-CAM has higher FSR values compared to those of LRP_T. The FSR scores show that G-CAM interpreter actually generates more incorrect explanations. This experiment allows us to conclude that G-CAM is the less reliable and the more vulnerable interpretation method of the two.

Model	FSR(%)	Location	
		LRP_T	G-CAM
VGG19	G-CAM	70.8	99.9
	LRP_T	97.8	40.6
ResNet50	G-CAM	86.2	99.9
	LRP_T	95.8	30.5
DenseNet121	G-CAM	67.4	98.8
	LRP_T	56.6	20.5

Table 9: The Fooling Success Rates (FSR) for Location fooling on three models with learning rate 1e-5 and penalty value 10. 10,000 images from ImageNet validation dataset are randomly selected to compute FSRs. The FSR table is to compare the performance of two interpreters with the same input fooling conditions.

7 Conclusions and Future Work

In this work, we replicate and extend the work of [4]. We show that interpretation methods can be fooled without significantly changing the accuracy. Furthermore, we provided evidence that increasing the learning rate and penalty terms both increase the FSR of a given model. Finally, we show that G-CAM is the less reliable interpretation method when compared to LRP_T, as G-CAM's generated interpretations have higher FSRs. It is important to note that for all experiments and extensions done in this work, the ending training and test accuracies do not drastically change.

Previously, we discuss how model interpretations vary with the learning rate and penalty term; however, the target layer is also an interesting hyperparameter to discover. In our experiments, we visualize the heatmaps on the last convolutional layer for VGG19 and the last block for ResNet50 and DenseNet121. However, in the future, one could analyze how the heatmaps and the corresponding FSRs change with target layer. *Transferability* is a key property of interpretation methods. For example, in this work, if we manipulate the model so that LRP is fooled, then the other interpretations also get fooled. The original work [4] shows that transferability exists in fooling methods. In the future, we could explore this idea more, and conduct rigorous experiments to ensure this is the case.

Contribution

Nikhil Krishna works on modifying and running codes and writing reports.
Chenqing Hua works on designing the experiments and writing reports.
Yiran Wang discusses the related work.

References

[1] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[4] Juyeon Heo, Sunghwan Joo, and Taesup Moon. Fooling neural network interpretations via adversarial model manipulation. *NeulPS Conference 2019*.

[5] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *Plos One*, 10(7), Oct 2015.

[6] 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. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

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

[8] Goodfellow, Pouget-Abadie Ian J., Jean, Mirza, Mehdi, Xu, David, Courville, Aaron, Yoshua, and et al. Generative adversarial networks, Jun 2014.

[9] Zaremba, Wojciech, Ilya, Bruna, Joan, Erhan, Dumitru, Goodfellow, Ian, Fergus, and et al. Intriguing properties of neural networks, Feb 2014.

[10] Jiawei Vasconcellos Su, Danilo Vasconcellos Vargas, and Sakurai Vasconcellos Kouichi. One pixel attack for fooling deep neural networks, Oct 2017.

[11] Adebayo, Julius, Gilmer, Justin, Michael, Goodfellow, Ian, Hardt, Moritz, Kim, and et al. Sanity checks for saliency maps, Oct 2018.

[12] Wojciech Samek, Alexander Binder, Sebastian Lapuschkin, and Klaus-Robert Muller. Understanding and comparing deep neural networks for age and gender classification. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2017.

[13] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, Nov 2015.