Improving Deep Learning Interpretability by Saliency Guided Training

Aya Abdelsalam Ismail, Héctor Corrada Bravo*, Soheil Feizi *
{asalam,sfeizi}@cs.umd.edu, hcorrada@gene.com
Department of Computer Science, University of Maryland

Abstract

Saliency methods have been widely used to highlight important input features in model predictions. Most existing methods use backpropagation on a modified gradient function to generate saliency maps. Thus, noisy gradients can result in unfaithful feature attributions. In this paper, we tackle this issue and introduce a *saliency guided training*† procedure for neural networks to reduce noisy gradients used in predictions while retaining the predictive performance of the model. Our saliency guided training procedure iteratively masks features with small and potentially noisy gradients while maximizing the similarity of model outputs for both masked and unmasked inputs. We apply the saliency guided training procedure to various synthetic and real data sets from computer vision, natural language processing, and time series across diverse neural architectures, including Recurrent Neural Networks, Convolutional Networks, and Transformers. Through qualitative and quantitative evaluations, we show that saliency guided training procedure significantly improves model interpretability across various domains while preserving its predictive performance.

1 Introduction

Deep Neural Networks (DNNs) have been widely used in a variety of different tasks [31, 26, 43, 37]; yet interpreting complex networks remains a challenge. Reliable explanations are necessary for critical domains like medicine, neuroscience, finance, and autonomous driving [9, 34]. Explanations are also useful for model debugging [63, 35]. As a result, various interpretability methods were developed to understand DNNs [5, 49, 22, 53, 52, 32, 51]. A common approach for understanding model decisions is to identify features in the input that highly influenced the final classification decision [6, 63, 53, 52, 47, 36, 64]. Such approaches, known as *saliency maps*, often use gradient calculations to assign an *importance* score to individual features, reflecting their influences on the model prediction.

Saliency methods aim to highlight meaningful input features in model predictions to humans; however, the maps produced are often noisy (i.e., contain visual noise). To improve the faithfulness of saliency maps, explanations methods that depend on more than one or higher-order gradient calculations were developed. For example, SmoothGrad [52] reduces saliency noise by adding noise to the input multiple times and then taking the average of the resulting saliency maps for each input. Integrated gradients [53], DeepLIFT [47] and Layer-wise Relevance Propagation [5] backpropagate through a modified gradient function [3] while Singla et al. [51] studies the use of higher-order gradients in saliency maps.

*Authors contributed equally

†Code: https://github.com/ayaabdelsalam91/saliency_guided_training

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

In this paper, we take a different approach to improve the interpretability of deep neural networks—instead of developing yet another saliency method, we propose a new *training procedure* that naturally leads to improved model explanations using current saliency methods. Our proposed training procedure, called *saliency guided training*, trains models that produce sparse, meaningful, and less noisy gradients without degrading model performance. This is done by iteratively masking input features with low gradient values (i.e., less important features) and then minimizing a loss function that combines (a) the KL divergence [27] between model outputs from the original and masked inputs, and (b) the appropriate loss function for the model prediction. This procedure reduces noise in model gradients without sacrificing its predictive performance.

To demonstrate the effectiveness of our proposed saliency guided training approach, we consider a variety of classification tasks for images, language, and multivariate time series across diverse neural architectures, including Convolutional Neural Networks (CNNs), Recurrent Neural Network (RNNs), and Transformers. In particular, we observe that using saliency guided training in image classification tasks leads to a reduction in visual saliency noise and sparser saliency maps, as shown in Figure 2. Saliency guided training also improves the comprehensiveness of the produced explanations for sentiment analysis, and fact extraction tasks as shown in Table 1.

In multivariate time series classification tasks, we observe an increase in the precision and recall of saliency maps when applying the proposed saliency guided training. Interestingly, we also find that the saliency guided training reduces the vanishing saliency issue of RNNs [20] as shown in Figure 6. Finally, we note that although we use the vanilla gradient for masking in the saliency guided training procedure, we observe significant improvements in the explanations produced after training by several other gradient-based saliency methods.

2 Background and Related Work

Interpretability is a rapidly growing area with several diverse lines of research. One strand of interpretability considers post-hoc explanation methods, aiming to explain why a trained model made a specific prediction for a given input. Post-hoc explanation methods can be divided into gradient-based methods [6, 53, 52, 47, 36, 46] that can be reformulated as computing backpropagation for a modified gradient function and perturbation-based approaches [63, 54, 42, 57] that perturb areas of the input and measure how much this changes the model output. Perkins et al. [40] uses gradients for feature selection through Grafting. Another line of works aims to measure the reliability of interpretability methods. This can be done by creating standardize benchmarks with interpretability metrics [18, 21, 11, 56, 45, 41] or debugging explanations [1, 24, 15, 2] by identifying test cases where explanations fail. Others [4, 12, 44, 61, 20] focus on modifying neural architectures for better interpretability. Similar to our line of work, Ghaeini et al. [14] and Ross et al. [44] incorporate explanations into the learning process. However, Ghaeini et al. [14] relies on the existence of the ground truth explanations while Ross et al. [44] relies on the availability of annotations about incorrect explanations for a particular input. Our proposed learning approach does not rely on such annotations; since most datasets only have ground truth labels, it may not be practical to assume the availability of positive or negative explanations.

Input level perturbation during training has been previously explored. [33, 19, 60, 50] use attention maps to improve segmentation for weakly supervised localization. Wang et al. [59] incorporates attention maps into training to improve classification accuracy. DeVries and Taylor [10] masks out square regions of input during training as a regularization technique to improve the robustness and overall performance of convolutional neural networks. Our work focuses on a different task which is increasing model interpretability through training in a self-supervised manner.

In this paper, we evaluate our learning procedure with the following saliency methods: **Gradient (GRAD)** [6] is the gradient of the output w.r.t the input. **Integrated Gradients (IG)** [53] calculates a path integral of the model gradient to the input from a non-informative reference point. **DeepLIFT (DL)** [47] compares the activation of each neuron to a reference activation; the relevance is the difference between the two activations. **SmoothGrad (SG)** [52] samples similar input by adding noise to the input and then takes the average of the resulting sensitivity maps for each sample. **Gradient SHAP (GS)** [36] adds noise to the input, then selects a point along the path between a reference point and input, and computes the gradient of outputs w.r.t those points.

We demonstrate the effectiveness of our training procedure using several neural network architectures: Convolution neural networks (CNNs) including VGG-16 [48], ResNet [16] and Temporal Convolu-

tional Network (TCN) [38, 28, 7], a CNN that handles sequences; Recurrent neural networks (RNNs) including LSTM [17] and LSTM with Input-Cell Attention [20]; as well as Transformers [58].

3 Notation

First, consider a classification problem on the input data $\{(X_i, y_i)\}_{i=1}^{n}$ such that each $X = [x_1, \ldots, x_N] \in \mathbb{R}^N$ has N features and y is the label. Let f_θ denote a neural network parameterized by θ. The standard training of the network involves minimizing the cross-entropy loss $\mathcal{L}$ over the training set as follows:

$$\underset{\theta}{\text{minimize}} \quad \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\left(f_\theta\left(X_i\right), y_i\right) \tag{1}$$

The gradient of the network output $f_\theta\left(X\right)$ with respect to the input X is given by $\nabla_X f_\theta\left(X\right)$. Let $S(.)$ be a sorting function such that $S_e(Z)$ is the eth smallest element in Z. Hence, $S\left(\nabla_X f_\theta\left(X\right)\right)$ is the sorted gradient. We define the input mask function $M_k(.)$ such that $M_k(S(X), X)$ replaces all x_i where $S(x_i) \in \{S_e\left(x_i\right)\}_{e=0}^{k}$ with a mask distribution, i.e., $M_k(S(X), X)$ removes the k lowest features from X based on the order provided by $S(X)$.

For a language input, we use $X = [x_1, \ldots, x_N]$ where $x_i \in \mathbb{R}^d$ is the feature embedding representing the ith word of the input. In that case, $S(X)$ would sort elements of X based on the sum of the gradient of the embeddings for each word x and $M_k(S(X), X)$ would mask the bottom k words according to that sorting. For a multivariate time series input, we use $X = [x_{1,1}, \ldots, x_{F,1}, \ldots, x_{F,T}] \in \mathbb{R}^{F \times T}$ where T is the number of time steps and F is the number of features per time step. $x_{i,t}$ is the input feature i at time t; sorting and masking would be done at the $x_{i,t}$ level.

For two discrete probability distributions P and Q defined on the same probability space $\mathcal{X}$, the Kullback–Leibler (KL) divergence [27] (or, relative entropy) from Q to P is given as D_{KL}:

$$D_{\text{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)}\right). \tag{2}$$

4 Saliency Guided Training

Existing gradient-based methods can produce noisy saliency maps as shown in Figure 1. The saliency map noise may be partially due to some uninformative local variations in partial derivatives. Using a standard training procedure based on ERM (expectation risk minimization), the gradient of the model w.r.t. the input (i.e., $\nabla_X f_\theta\left(X\right)$) may fluctuate sharply via small input perturbations [52].

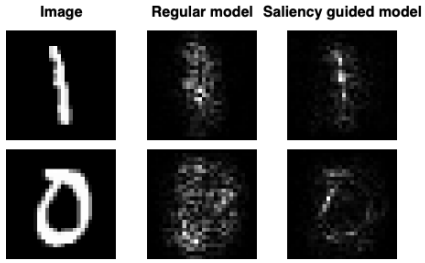

Figure 1: Saliency maps produced by typical training versus saliency guided training.

If gradient-based explanation methods faithfully interpret the model's predictions, irrelevant features should have gradient values close to zero. Building on this intuition, we introduce *saliency guided training*, a procedure to train neural networks such that input gradients computed from trained models provide more faithful measures to downstream (gradient-based) saliency methods. Saliency guided training aims to reduce gradient values of irrelevant features without sacrificing the model performance. During saliency guided training, for every input X, we create a new input $\widetilde{X}$ by masking the features with low gradient values as follows:

$$\widetilde{X} = M_k(S(\nabla_X f_\theta\left(X\right)), X) \tag{3}$$

$\widetilde{X}$ is then passed through the network which results in an output $f_\theta(\widetilde{X})$. In addition to the classification loss, the saliency guided training minimizes the KL divergence between $f_\theta(X)$ and $f_\theta(\widetilde{X})$ to ensure that the trained model produces similar output probability distributions over labels for both masked and unmasked inputs. The optimization problem for the saliency guided training is:

$$\underset{\theta}{\text{minimize}} \ \frac{1}{n} \sum_{i=1}^{n} \left[\mathcal{L}\Big(f_\theta\left(X_i\right), y_i \Big) + \lambda D_{KL}\Big(f_\theta\left(X_i\right) \parallel f_\theta(\widetilde{X_i}) \Big) \right] \tag{4}$$

where λ is a hyperparameter to balance between the cross-entropy classification loss and the KL divergence term. Since this loss function is differentiable with respect to θ, it can be optimized using existing gradient-based optimization methods. The KL divergence term encourages the model to produce similar outputs for the original input X and masked input $\widetilde{X}$. For this to happen, the model will need to learn to assign low gradient values to irrelevant features in model predictions. This potentially results in sparse and more faithful gradients as shown in Figure 1.

Masking functions: In images and time series data, features with low gradients are replaced with random values within the feature range. In language tasks, the masking function replaces the low salient word with the previous high salient word. This allows us to emphasize on high salient words and remove non-salient ones while maintaining the sentence length. The selection of k is dataset-dependent. It depends on the amount of irrelevant information in a training sample. For example, since most pixels in MNIST are uninformative, a larger k is desired. Detailed hyperparameters used is available in the appendix. Note that, only input features are masked during the saliency guided training.

Limitations: (a) Compared to traditional training, our proposed training procedure is more computationally expensive. Specifically, the memory needed is doubled since now in addition to storing the batch, we are storing the masked batch as well. Similar to adversarial training, this training process is slow and takes a larger number of epochs to converge. For example, the standard training of a CIFAR-10 model usually takes on average 118 epochs to converge where each epoch is roughly 24 seconds. Using the saliency guided training, the convergence takes about 124 epochs where each epoch takes roughly 75 seconds (all experiments on the same GPU). (b) Our training procedure requires two hyperparameters k and λ which might require a hyperparameter search (we find that $\lambda = 1$ works well in all of our experiments).

Algorithm 1: Saliency Guided Training

Given: Training samples X, # of features to be masked k, learning rate τ, hyperparameter λ

Initialize f_θ

for $i \leftarrow 1$ **to** *epochs* **do**

  for *minibatch* **do**

    Compute the masked input:

      Get sorted index I for the gradient of output with respect to the input.

      $I = S\Big(\nabla_X f_{\theta_i}\left(X\right)\Big)$

      Mask bottom k features of the original input.

      $\widetilde{X} = M_k(I, X)$

    Compute the loss function:

      $L_i = \mathcal{L}\Big(f_{\theta_i}\left(X\right), y \Big) + \lambda D_{KL}\Big(f_{\theta_i}\left(X\right) \parallel f_{\theta_i}(\widetilde{X}) \Big)$

    Use the gradient to update network parameters:

      $f_{\theta_{i+1}} = f_{\theta_i} - \tau \, \nabla_{\theta_i} L_i$

  end

end

5 Experiments

All experiments have been repeated 5 times; the results reported below are the average of the 5 runs. Hyperparameters used for each experiment, along with the standard error bars and details on computational resources are available in supplementary materials.

5.1 Saliency Guided Training for Images

In the following section, we compare gradient-based explanations produced by regular training versus saliency guided training for MNIST [30] trained on a simple CNN [29], for CIFAR10 [25] trained on ResNet18 [16] and for BIRD [13] trained on VGG-16 [48]. Further details about the datasets and models are available in the supplementary material.

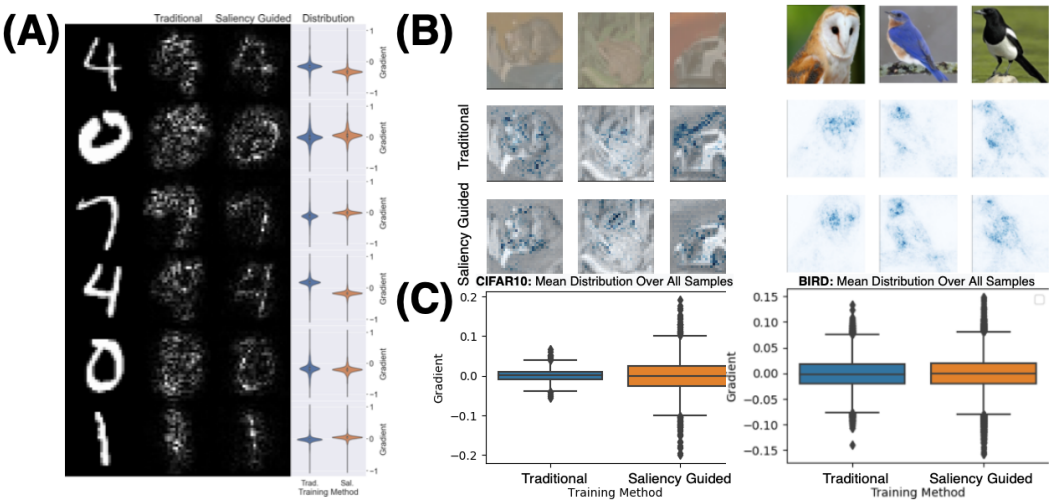

Figure 2: (A) Comparison between different training methods on MNIST along with distributions of gradient values in each sample. (B) Saliency maps for CIFAR10 and BIRD datasets using regular and saliency guided training. (C) Distribution of gradient means across examples. Maps produced by saliency guided training are more precise: most features have gradient values around zero with large gaps between mean and outliers. Here gradients around zero indicate uninformative features, while very large and very small gradients indicate informative features. Saliency guided training helps reduce noisy fluctuating gradients in between as shown in the box plots.

Quality of Saliency Maps for Images

For an image classification problem, in many cases, most features are redundant and not needed by the model to make the prediction. Consider the background of an object in an image; although it covers most of the image, backgrounds are often not essential in the classification task. If the model is focusing on the object rather than the background, we would want the background gradient (i.e., most of the features) to be close to zero.

The examples shown in Figure 2 were correctly classified by both models. Gradients are scaled per sample to have values between -1 and 1. In Figure 2 (A) and Figure 2 (B), saliency maps produced by a model trained with saliency guided training were more precise than that trained traditionally. Most saliency maps produced by saliency guided training highlight the object itself rather than the background across different datasets. The distributions of gradient values per sample in Figure 2 (A) show that most features have small gradient values (near zero) with a large separation of high salient features away from zero for the saliency guided training. Similarly, in Figure 2 (C), we find that over the entire dataset, gradient values produced by the saliency guided training tend to be concentrated around zero with a large separation between the mean and outliers (highly salient features), indicating the model's ability to differentiate between informative and non-informative features.

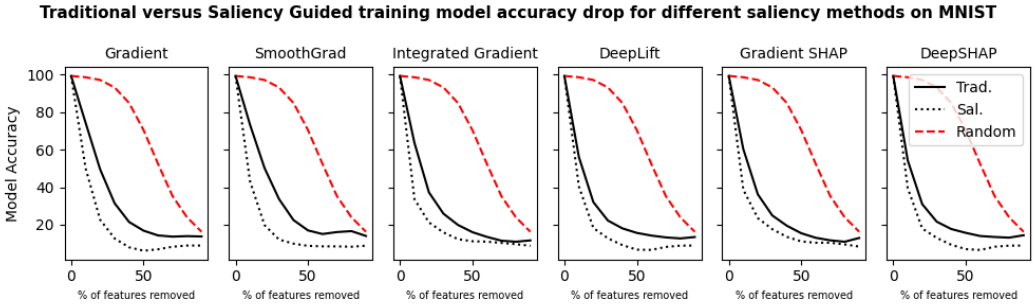

Figure 3: Model accuracy drop when removing features with high saliency using traditional and the saliency guided training for different gradient-based methods against a random baseline. A steeper drop indicates a better performance. We find that regardless of the saliency method used, the performance improves by the saliency guided training.

Model Accuracy Drop

We compare the saliency guided training and traditional training for different saliency methods with modification-based evaluation [45, 41, 23]: First, features are ranked according to the saliency values. Then, higher-ranked features are recursively eliminated (the original background in MNIST replaces the eliminated features). Finally, the degradation to the trained model accuracy is reported. This is done at different feature percentages. A steeper drop indicates that the removed features affected the model accuracy more. Figure 3 compares the model performance degradation on different gradient-based methods; the saliency guided training shows a steeper accuracy drop regardless of the saliency method used.

This experiment can only be performed on a dataset like MNIST since the uninformative feature distribution is known (black background), while this is not the case in other datasets that we have considered. Although such modification-based evaluation methods have been applied to other datasets, [45, 41, 23]; Hooker et al. [18] showed that removing features produces samples from a different data distribution violating the underlying IID assumption (i.e., the training and evaluation data come from identical distributions). When the feature replacement comes from a different distribution, it is unclear whether the degradation in the model performance is from the distribution shift or the removal of informative features. For that reason, we need to make sure that the model is trained on the mask used during testing to avoid this undesired effect.

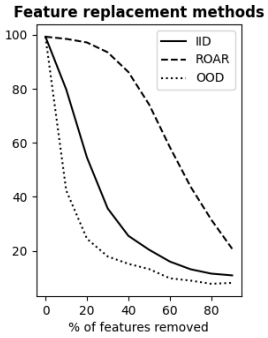

Figure 4: Accuracy drop in different modification-based evaluation masking approaches.

Hooker et al. [18] proposes ROAR where the model is retrained after the feature elimination. However, due to the data redundancy, the retrained model can rely on different features to achieve the same accuracy. Figure 4 shows the model accuracy drop on traditionally trained MNIST when removing the salient features. The IID line represents replacing features with the black MNIST background (known uninformative distribution), which acts as the ground truth in this particular dataset. The OOD line represents replacing the features with the mean image pixel value as done by [45, 41, 23]; and ROAR shows replacing features with the mean value and retraining the model as proposed by Hooker et al. [18]. Since neither OOD nor ROAR produce results similar to those produced by the IID feature replacement, we argue that modification-based evaluation methods may provide unreliable results unless the uninformative IID distribution is known. We leave further exploration of modification-based evaluation methods to future work.

5.2 Saliency Guided Training for Language

We compare the interpretability of recurrent models trained on language tasks using the ERASER [11] benchmark. ERASER was designed to capture how well an explanation provided by models aligns

with human rationales and how faithful these explanations are (i.e., the degree to which explanation influences the predictions). For our purpose, we only focus on the faithfulness of the explanations.

ERASER provides two metrics to measure interpretability. *Comprehensiveness* evaluates if all features needed to make a prediction are selected. To calculate an explanation comprehensiveness, a new input $\overline{X}_i$ is created such that $\overline{X}_i = X_i - R_i$ where R_i is predicted rationales (i.e. the words selected by saliency method as informative). Let $f_\theta\left(X_i\right)_j$ be the prediction of model for class j. The model comprehensiveness is calculated as:

$$\text{Comprehensiveness} = f_\theta\left(X_i\right)_j - f_\theta\left(\overline{X}_i\right)_j \tag{5}$$

A high score here implies that the explanation removed was influential in the predictions. The second metric is *Sufficiency* that evaluates if the extracted explanations contain enough signal to make a prediction. The following equation gives the explanation sufficiency:

$$\text{Sufficiency} = f_\theta\left(X_i\right)_j - f_\theta\left(R_i\right)_j \tag{6}$$

A lower score implies that the explanations are adequate for a model prediction. The comprehensiveness and sufficiency were calculated at different percentages of features (similar to [11] percentages are $1\%, 5\%, 10\%, 20\%$ and 50%), and Area Over the Perturbation Curve (AOPC) is reported.

We focus on datasets that can be formulated as a classification problem: *Movie Reviews:* [62] positive/negative sentiment classification for movie reviews. *FEVER:* [55] a fact extraction and verification dataset where the goal is verifying claims from textual sources; each claim can either be supported or refuted. *e-SNLI:* [8] a natural language inference task where sentence pairs are labeled as entailment, contradiction, neutral and, supporting.

Word embeddings are generated from Glove [39]; then passed to a bidirectional LSTM [17] for classification. Table 1 compares the scores produced by different saliency methods for traditional and saliency guided training against random assignment baseline. We found that saliency guided training results in a significant improvement in both comprehensiveness and sufficiently for sentiment analysis task *Movie Reviews* dataset. While for fact extraction task *FEVER* dataset, and natural language inference task *e-SNLI* dataset saliency guided training improves comprehensiveness and there is no obvious improvement in sufficiency (this might be due to the adversarial effect of shrinking the sentence to a much smaller size since the number of words identified as "rationales" is smaller than the remaining words).

	Gradient		Integrated Gradient		SmoothGrad		Random
	Trad.	Sal. Guided	Trad.	Sal. Guided	Trad.	Sal. Guided	
Movies							
Comprehensiveness ↑	0.200	*0.240*	0.265	***0.306***	0.198	*0.256*	0.056
Sufficiency ↓	0.042	*0.013*	0.054	***0.002***	0.034	*0.008*	0.294
FEVER							
Comprehensiveness↑	0.007	*0.008*	0.008	***0.009***	0.007	*0.008*	0.001
Sufficiency↓	0.012	*0.011*	0.005	*0.004*	0.006	0.006	***0.003***
e-SNLI							
Comprehensiveness ↑	0.117	***0.126***	0.099	*0.104*	0.117	*0.118*	0.058
Sufficiency↓	0.420	*0.387*	0.461	*0.419*	0.476	*0.455*	***0.366***

Table 1: Eraser benchmark scores: *Comprehensiveness* and *sufficiency* are in terms of AOPC. 'Random' is a baseline when words are assigned random scores.

5.3 Saliency Guided Training for Time Series

We evaluated saliency guided training for multivariate time series, both quality on multivariate time series MNIST and quantitatively through synthetic data.

Saliency Maps Quality for Multivariate Time Series

We compare the saliency maps produced on MNIST treated as a multivariate time series where one image axis is time. Figure 5 shows the saliency maps produced by different *(neural architecture, saliency method)* pairs when different training procedures were used. There is a visible improvement in saliency quality across different networks when saliency guided training is used.

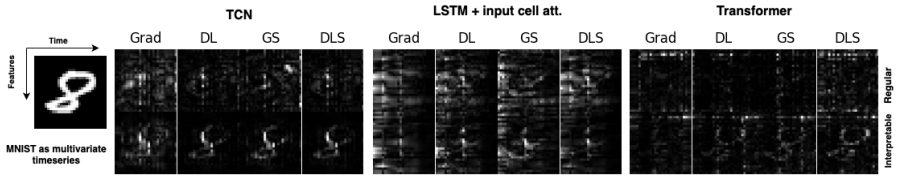

Figure 5: Saliency maps produced for *(neural architecture, saliency method)* pairs. Traditional training was used for networks in the 1st row, while saliency guided training was used for the 2nd row. Grad, DL, GS and DLS stand for Gradient, DeepLift, Gradient SHAP and DeepSHAP, respectively. There is an improvement in the quality of saliency maps when saliency guided training is used.

Quantitative Analysis on Synthetic Data

We evaluated the saliency guided training on a multivariate time series benchmark proposed by Ismail et al. [21]. The benchmark consists of 10 synthetic datasets, each examining different design aspects in typical time series datasets. Informative features are highlighted by the addition of a constant μ to the positive class and subtraction of μ from the negative class. Following Ismail et al. [21], we compare 4 neural architectures: LSTM [17], LSTM with Input-Cell Attention [20], Temporal Convolutional Network (TCN) [28] and, Transformers [58]. Additional details about the dataset and architectures are provided in the supplementary material.

Quantitatively measuring the interpretability of a *(neural architecture, saliency method)* pair involves applying the saliency method, ranking features according to the saliency values, replacing high salient features with uninformative features from the original distribution at different percentages. Finally, the area under the precision curve (AUP) and the area under the recall curve (AUR) is calculated by the precision/recall values at different levels of degradation. Similar to Ismail et al. [21], we compare the AUP and AUR with a random baseline; since the baseline might be different for different models, we reported the difference between metrics values generated using the saliency method and the baseline. For example, the difference between gradient and random baseline *Diff*(AUP) when the model is trained traditionally is calculated as:

$$\textit{Diff}(\text{AUP})_{Grad,Trad.} = AUP_{Grad,Trad.} - AUP_{Random,Trad.} \tag{7}$$

Similarly difference when the model is trained using saliency guided training is:

$$\textit{Diff}(\text{AUP})_{Grad,Sal.} = AUP_{Grad,Sal.} - AUP_{Random,Sal.} \tag{8}$$

The mean metrics over all 10 datasets is shown in Table 2. Higher values indicate better performance; negative values indicate performance similar to random feature assignment. Overall, the best performance was achieved by *(TCN, Integrated gradients)* when using saliency guided training. Detailed results for each dataset are available in the supplementary material.

Metric	Architecture	Gradient		Integrated Gradient		DeepLIFT		Gradient SHAP		DeepSHAP		SmoothGrad	
		Trad.	Sal.	Trad.	Sal.	Trad.	Sal.	Trad.	Sal.	Trad.	Sal.	Trad.	Sal.
Diff(AUP)	LSTM	-0.113	-0.119	-0.083	-0.024	-0.097	-0.108	-0.088	-0.069	-0.098	-0.109	-0.110	-0.097
	LSTM + Input.	0.060	*0.118*	0.188	*0.245*	0.202	*0.263*	0.198	*0.250*	0.214	***0.272***	0.040	*0.084*
	TCN	0.106	*0.168*	0.233	***0.291***	0.248	*0.270*	0.235	*0.288*	0.263	*0.280*	0.088	*0.155*
	Transformer	-0.054	-0.062	*0.061*	0.044	-0.040	-0.032	***0.069***	0.023	-0.014	-0.055	-0.018	-0.046
Diff(AUR)	LSTM	-0.017	*0.019*	0.062	***0.121***	0.047	*0.089*	0.060	*0.102*	0.031	*0.075*	*0.007*	0.004
	LSTM + Input.	0.075	*0.136*	0.185	*0.198*	0.187	***0.204***	0.182	*0.196*	0.183	*0.201*	0.043	*0.111*
	TCN	0.125	*0.171*	0.191	***0.210***	0.202	*0.204*	0.185	*0.209*	*0.196*	0.192	0.046	*0.138*
	Transformer	0.102	*0.104*	***0.182***	0.176	0.145	*0.146*	*0.171*	0.162	*0.101*	0.065	*0.040*	0.018

Table 2: The mean difference in weighted AUP and AUR for different *(neural architecture, saliency method)* pairs. Overall, the best preference was achieved by TCN when using Integrated gradients as a saliency method and saliency guided training procedure.

Saliency Guided Training reduces vanishing saliency of recurrent neural networks

Ismail et al. [20] showed that saliency maps in RNNs vanish over time, biasing detection of salient features only to later time steps. This section investigates if using saliency guided training reduces the vanishing saliency issue in RNNs. Repeating experiments done by Ismail et al. [20], three

synthetic datasets were generated as shown Figure 6 (A). The specific features and the time intervals (boxes) on which they are considered important are varied between datasets to test the model's ability to capture importance at different time intervals. We trained an LSTM with traditional and saliency guided training procedures.

The area under precision curve (AUP) and the area under the recall curve (AUR) are calculated by the precision/recall values at different levels of degradation. Higher AUP and AUR suggest better performance. Results are shown in Figure 6 (B).

A traditionally trained LSTM shows clear bias in detecting features in the later time steps; AUP and AUR increase as informative features move to later time steps. When saliency guided training is used, LSTM was able to identify informative features regardless of their locations in time.

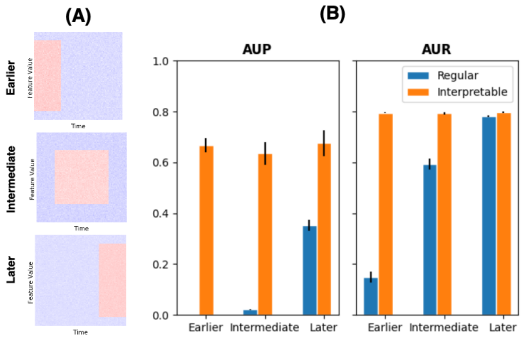

Figure 6: (A) Samples from 3 different simulated datasets, informative features are located at the earlier, intermediate, and later time steps. (B) AUP and AUR were produced by LSTM by traditional and saliency guided training procedures. Traditionally trained LSTM shows clear bias in detecting features in the later time steps. When saliency guided training is used, there is no time bias.

6 Summary and Conclusion

We propose *saliency guided training* as a new training procedure that improves the quality of explanations produced by existing gradient-based saliency methods. *saliency guided training* is optimized to reduce gradient values for irrelevant features. This is done by masking input features with low gradients and then minimizing the KL divergence between outputs from the original and masked inputs along with the main loss function. We demonstrated the effectiveness of the *saliency guided training* on images, language, and multivariate time series.

Our proposed training method encourages models to sharpen the gradient-based explanations they provide. It does this however without requiring explanations as input. It instead may be cast as a regularization procedure where regularization is provided by feature sparsity driven by a gradient-based feature attribution. This is an alternative approach to using ground truth explanations to force the model to be *right for the right reasons* [44]. We found that training model explanations in an unsupervised fashion also improves model faithfulness. This opens an interesting avenue for other unsupervised, perhaps regularization-based, methods to improve the interpretability of prediction models.

7 Acknowledgments

This project was supported in part by NSF CAREER AWARD 1942230, a grant from NIST 60NANB20D134, NSF award CDSE:1854532, ONR grant 13370299 and AWS Machine Learning Research Award.

References

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems*, 2018.

[2] Julius Adebayo, Michael Muelly, Ilaria Liccardi, and Been Kim. Debugging tests for model explanations. *arXiv preprint arXiv:2011.05429*, 2020.

[3] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *International Conference on Learning Representations*, 2018.

[4] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *Advances in Neural Information Processing Systems*, 2014.

[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. In *PLoS ONE*, 2015.

[6] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. How to explain individual classification decisions. In *Journal of Machine Learning Research*, 2010.

[7] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[8] Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-snli: Natural language inference with natural language explanations. *arXiv preprint arXiv:1812.01193*, 2018.

[9] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *International conference on knowledge discovery and data mining*, 2015.

[10] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[11] 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.

[12] Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

[13] Gerry. 265 bird species, 2021. URL `https://www.kaggle.com/gpiosenka/100-bird-species/`.

[14] Reza Ghaeini, Xiaoli Z Fern, Hamed Shahbazi, and Prasad Tadepalli. Saliency learning: Teaching the model where to pay attention. *arXiv preprint arXiv:1902.08649*, 2019.

[15] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *AAAI Conference on Artificial Intelligence*, 2019.

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[17] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. In *Neural computation*, 1997.

[18] Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. A benchmark for interpretability methods in deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

[19] Qibin Hou, Peng-Tao Jiang, Yunchao Wei, and Ming-Ming Cheng. Self-erasing network for integral object attention. *arXiv preprint arXiv:1810.09821*, 2018.

[20] Aya Abdelsalam Ismail, Mohamed Gunady, Luiz Pessoa, Hector Corrada Bravo, and Soheil Feizi. Input-cell attention reduces vanishing saliency of recurrent neural networks. In *Advances in Neural Information Processing Systems*, 2019.

[21] Aya Abdelsalam Ismail, Mohamed Gunady, Héctor Corrada Bravo, and Soheil Feizi. Benchmarking deep learning interpretability in time series predictions. *arXiv preprint arXiv:2010.13924*, 2020.

[22] Pieter-Jan Kindermans, Kristof Schütt, Klaus-Robert Müller, and Sven Dähne. Investigating the influence of noise and distractors on the interpretation of neural networks. *arXiv preprint arXiv:1611.07270*, 2016.

[23] Pieter-Jan Kindermans, Kristof T Schütt, Maximilian Alber, Klaus-Robert Müller, Dumitru Erhan, Been Kim, and Sven Dähne. Learning how to explain neural networks: Patternnet and patternattribution. *arXiv preprint arXiv:1705.05598*, 2017.

[24] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. The (un) reliability of saliency methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, 2019.

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Citeseer*, 2009.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 2012.

[27] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 1951.

[28] Colin Lea, Michael Flynn, Rene Vidal, Austin Reiter, and Gregory Hager. Temporal convolutional networks for action segmentation and detection. In *Conference on Computer Vision and Pattern Recognition*, 2017.

[29] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[30] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database, 2010.

[31] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 2015.

[32] Alexander Levine, Sahil Singla, and Soheil Feizi. Certifiably robust interpretation in deep learning. *arXiv preprint arXiv:1905.12105*, 2019.

[33] Kunpeng Li, Ziyan Wu, Kuan-Chuan Peng, Jan Ernst, and Yun Fu. Tell me where to look: Guided attention inference network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[34] Zachary C Lipton. The mythos of model interpretability. In *Queue*, 2018.

[35] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012.

[36] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017.

[37] Ziad Obermeyer and Ezekiel J Emanuel. Predicting the future—big data, machine learning, and clinical medicine. In *The New England journal of medicine*, 2016.

[38] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[39] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014.

[40] Simon Perkins, Kevin Lacker, and James Theiler. Grafting: Fast, incremental feature selection by gradient descent in function space. *The Journal of Machine Learning Research*, 2003.

[41] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.

[42] 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*, 2016.

[43] Michael L Rich. Machine learning, automated suspicion algorithms, and the fourth amendment. In *University of Pennsylvania Law Review*, 2016.

[44] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. Right for the right reasons: Training differentiable models by constraining their explanations. *arXiv preprint arXiv:1703.03717*, 2017.

[45] Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 2016.

[46] 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*, 2017.

[47] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International Conference on Machine Learning*, 2017.

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

[49] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, 2013.

[50] Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *2017 IEEE international conference on computer vision (ICCV)*. IEEE, 2017.

[51] Sahil Singla, Eric Wallace, Shi Feng, and Soheil Feizi. Understanding impacts of high-order loss approximations and features in deep learning interpretation. In *International Conference on Machine Learning*. PMLR, 2019.

[52] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smooth-grad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[53] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, 2017.

[54] Harini Suresh, Nathan Hunt, Alistair Johnson, Leo Anthony Celi, Peter Szolovits, and Marzyeh Ghassemi. Clinical intervention prediction and understanding using deep networks. *arXiv preprint arXiv:1705.08498*, 2017.

[55] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*, 2018.

[56] 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*, 2020.

[57] Sana Tonekaboni, Shalmali Joshi, Kieran Campbell, David Duvenaud, and Anna Goldenberg. What went wrong and when? instance-wise feature importance for time-series models. *arXiv preprint arXiv:2003.02821*, 2020.

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[59] Lezi Wang, Ziyan Wu, Srikrishna Karanam, Kuan-Chuan Peng, Rajat Vikram Singh, Bo Liu, and Dimitris N Metaxas. Sharpen focus: Learning with attention separability and consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

[60] Yunchao Wei, Jiashi Feng, Xiaodan Liang, Ming-Ming Cheng, Yao Zhao, and Shuicheng Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[61] Mike Wu, Michael C Hughes, Sonali Parbhoo, Maurizio Zazzi, Volker Roth, and Finale Doshi-Velez. Beyond sparsity: Tree regularization of deep models for interpretability. In *AAAI Conference on Artificial Intelligence*, 2018.

[62] Omar Zaidan and Jason Eisner. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 conference on Empirical methods in natural language processing*, 2008.

[63] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, 2014.

[64] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Checklist

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above, along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] **section 4**
   (c) Did you discuss any potential negative social impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] **In the supplemental material**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] **In the supplemental material**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] **In the supplemental material**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL?[N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]