An Adversarial Approach for Explaining the Predictions of Deep Neural Networks

Arash Rahnama
Modzy
arash.rahnama@modzy.com

Andrew Tseng
Modzy
andrew.tseng@modzy.com

Abstract

Machine learning models have been successfully applied to a wide range of applications including computer vision, natural language processing, and speech recognition. A successful implementation of these models however, usually relies on deep neural networks (DNNs) which are treated as opaque black-box systems due to their incomprehensible complexity and intricate internal mechanism. In this work, we present a novel algorithm for explaining the predictions of a DNN using adversarial machine learning. Our approach identifies the relative importance of input features in relation to the predictions based on the behavior of an adversarial attack on the DNN. Our algorithm has the advantage of being fast, consistent, and easy to implement and interpret. We present our detailed analysis that demonstrates how the behavior of an adversarial attack, given a DNN and a task, stays consistent for any input test data point proving the generality of our approach. Our analysis enables us to produce consistent and efficient explanations. We illustrate the effectiveness of our approach by conducting experiments using a variety of DNNs, tasks, and datasets. Finally, we compare our work with other well-known techniques in the current literature.

1 Introduction

Explaining the outcomes of complex machine learning models is a perquisite for establishing trust between the machines and users. As humans increasingly rely on DNNs to processing large amounts of data and make decisions, it is crucial to develop solutions that can interpret the predictions of DNNs in a user-friendly manner. Explaining the outcomes of a model can help reduce bias and contribute to improvements in model design, performance, and accountability by providing beneficial insights into how models behave [1]. Consequently, the field of explainable artificial intelligence systems, XAI, has gained traction in recent years, where researchers from different disciplines have come together to define, design and evaluate explainable systems [2–4]. The majority of current explainability algorithms for DNNs produce an explanation for a single input-output pair: an input data point fed into the DNN and the respective prediction made by the DNN. The algorithm usually finds the most important features in the input contributing the most to the model's predictions and selects those as explanations for the model's behavior [5]. The majority of these algorithms find the important features using either a *perturbation-based* approach or a *saliency-based* approach [6]. The saliency-based approaches rely on gradients of the outputs in relation to the inputs to find the important features [7, 8]. Perturbation-based methods on the other hand apply small local changes to the input, track the changes in the output, and find and rank the important input features [9, 10].

One main problem with current state-of-the-art explainability tools is their reliance on a large set of hyper-parameters. This leads to local instability of explanations and can negatively affect the user's experience [5]. An explainability algorithm should satisfy 3 properties: 1- It has to produce human-understandable explanations, 2- It has to be locally consistent and efficient, 3- It should be

Preprint. Under review.

user-friendly, easy to apply and quick in providing explanations. In this work, we propose a new algorithm, explanations via adversarial attacks, which satisfies these 3 important properties and more. We call our method **A**dversarial **E**xplainations for **A**rtificial **I**ntelligence systems or AXAI [1]. AXAI inherits from the nature of adversarial attacks to automatically find and select important features affecting the model's prediction to produce explanations. The idea behind our work comes from the natural behavior of adversarial attacks. The attacks tend to manipulate important features in the input to deceive a DNN. The logic is simple, rather than trying to build a model that learns to explain the DNN's behavior, why don't we utilize the nature of attacks to learn this behavior? One who knows how to fool a model, certainly knows what the model may be thinking. Another benefit of our approach is that certain attacks, such as the Projected Gradient Descent (PGD) method [11], are fast, efficient, and consistent in their adversarial behavior. Our work further aims to solve at least 2 problems: 1- Provide fast explanations without a need for model training, 2- Reduce the need for selecting a large set of hyper-parameters to produce consistent results.

Obviously, one needs to first show how adversarial attacks link to explainbility, i.e., how an attack can point to the important features in the input and how one can filter out the unimportant ones to produce explanations. Further, one needs to show how an adversary behaves similarly in its approach across models, tasks and datasets so that the explanations are consistent, stable, and applicable to a large group of models. Here, we present a novel algorithm for explaining the DNN's predictions in multiple domains including text, audio and image. In particular, this paper makes the following contributions:

- We show that given an ℓ_2 PGD attack and a trained DNN, the distribution of attack magnitudes vs. frequency across all unseen test inputs follows a beta distribution, regardless of the task and dataset. We also show that these distributions are symmetric and the differences between their means, medians, and quantiles are not statistically significant.
- We show that the most important input features, i.e., features with the largest effect on the model's predictions, can be found using a consistent rule across different DNN architectures, datasets, and tasks. This rule leverages the properties of the distributions explained above.
- We propose a novel algorithm for explaining the outcomes of DNNs and provide a detailed analysis of our algorithm's performance for different DNN architectures, datasets and tasks.
- We benchmark our algorithm against methods such as LIME and SHAP [6, 9] and show that our algorithm performs faster while producing similar or better explainability results.

2 Related Work

One of the popular explainability solutions called LIME [9] assumes that DNNs are linear locally. LIME trains weighted linear models on the top of the DNN for perturbed samples around a target input to produce explanations. The computational bottleneck in LIME is caused by the training part where a selected number of perturbed samples are sent through the DNN for learning the explanation. Certain combination of LIME's hyper-parameters can produce unstable results [5]. DeepLIFT produces explanations by modeling the slope of gradient changes of output with respect to the input [12]. Grad-CAM is a saliency-based method that uses the gradients of the input at the final convolutional layer to produce coarse localization maps pointing to important regions in the input [8]. The majority of approaches based on sensitivity maps fail to produce explanations that only rely on important features. Creators of DeepLIFT associate this lack of stability to the behavior of activation functions such as ReLU. [13] proposed Smooth Grad which uses gradients and Gaussian based de-noising methods to produce stable explanations. The authors of the paper mention that large outlier values in the gradient maps produced by gradient differentiation may cause instability. In our algorithm, we overcome the problem of instability by utilizing the density of attacks, which are created iteratively on segments. Some other important works in this area are given in [14–20].

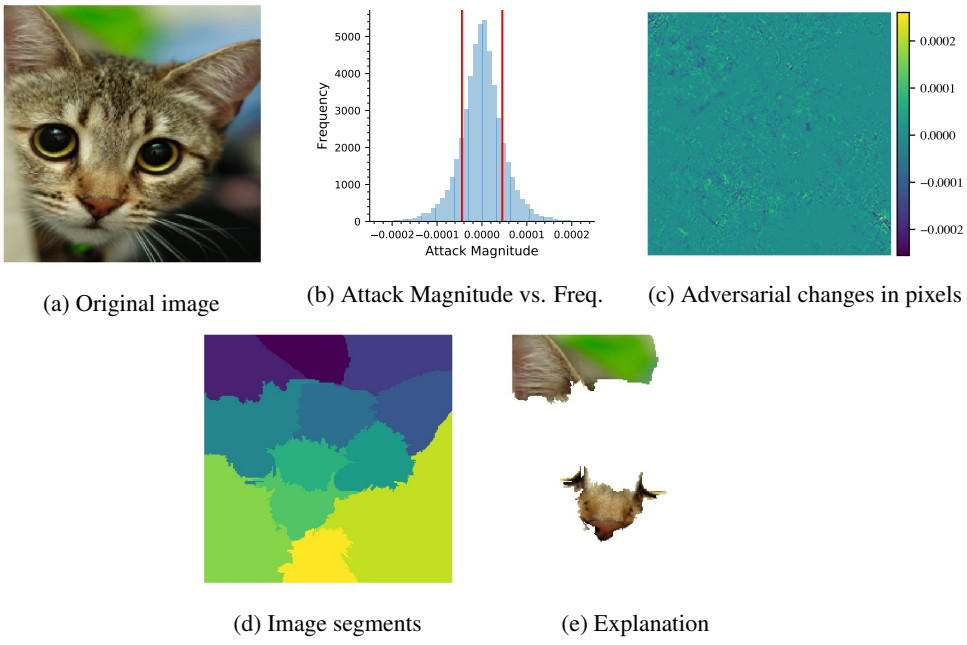
DNNs are vulnerable to subtle adversarial perturbations applied to their input. The basic idea behind most adversarial attacks revolves around solving a maximization problem with a constraint that keeps the distance between the original input and adversarial input small, so that the adversarial input, while capable of fooling the DNN, is not perceptually recognizable by humans. The connection between model interpretation and attacks has recently gravitated the interest of researchers. [21] and [22]

[1] Code will be readily available.

showed that one benefit of adversarial examples is that they reveal useful insights into the salient features of input data and their effects on DNNs' predictions. Our solution relies on the nature of adversarial attacks to select and produce important and explainable features given a specific input and DNN. Our work puts more emphasis on model interpretability, where we make use of the information obtained from an adversarial attack on a DNN to de-noise the sensitivity maps and produce stable explanations. We de-noise the gradient map by utilizing the iterative nature of the PGD attack and by considering only a minimum number of highly influential gradients that contribute the most to the predictions. We use the density of gradients in a number of segments to remove the noise that was not filtered out in the previous steps and produce human-interpretable explanations.

3 Main Results

The core idea behind our approach, AXAI, is to utilize the knowledge gained from an adversarial attack on a DNN and an input, to find the important features in the input in order to produce good explanations. This is done by mapping "carefully filtered attacked inputs" onto predefined segments and filtering out the unimportant features. This will be discussed in more detail in later sections. First let's look at an example in Fig. 1 to see how our approach works. Given an image classification DNN, the ℓ_2 adversarial attack changes the pixels in the entire image, as seen in Fig. 1c. The reason for this is simple: each pixel value is changed by the adversary so that the accumulated loss value can increase enough to fool the DNN. Fig. 1b shows the distribution of the attack on this image. The x-axis represents the magnitude of the pixel changes and the y-axis represents the number of pixels given each value on x-axis. AXAI maps the strongly attacked pixels to the image segments of the original image and filters out the segments with highest density of attacked pixels which meet certain criteria to produce explanations. Fig. 1c shows the value changes for the important attacked pixels. As we will show, the important features used for explanations are located at specific sections in the tails of the distribution given in Fig. 1b. These are the pixels that directly affect the classification decision made by the model. We use QuickShift [23] for segmenting the input image (Fig. 1d). Fig. 1e shows the explanation produced by our algorithm.



(a) Original image (b) Attack Magnitude vs. Freq. (c) Adversarial changes in pixels

(d) Image segments (e) Explanation

Figure 1: A simple example depicting the steps taken in AXAI to produce explanations.

Algorithm 1 details the steps taken by AXAI to produce an explanation E for the output of a selected model f. Suppose that input X is segmented into p groups using a segmentation method and that the attack magnitudes for the input X and DNN f are obtained. Let X_{diff} be the difference between the original X and adversarial X'. We filter out the low intensity attack magnitudes X_{diff} and create a

Boolean array X_{difft}, where values larger than a threshold, are only set to True. Let Su be the set of unique segments, $Su = \{Su_1, \ldots, Su_p\}$. Next, we map the filtered attack X_{difft} to the segments Su, and create a new list of filtered attack groups, $Su_x = \{Su_{x_1}, \ldots, Su_{x_p}\}$. The mapping function, *Map* in Algorithm 1, simply stacks the filtered attacks on the segments and groups the filtered attack X_{difft} based on the segments. Finally, the attack density of each unique segment can be written as $Su_d = \{\frac{card(Su_{x_1})}{card(Su_1)}, \ldots, \frac{card(Su_{x_p})}{card(Su_p)}\}$ (*Calculate_density* in Algorithm 1). We then extract the indices j's of the top K maximum values in Su_d (*TopK_indices* in Algorithm 1), and produce $Su(j)$ as explanation E for the input X. In next sections, we explain each step in details.

Algorithm 1 AXAI

Require: Model f, input X

$X' \leftarrow Attack(f, X)$ ▷ i.e. PGD attack

$X_{diff} = x' - x$ ▷ The attack magnitudes

$X_{difft} \leftarrow Threshold(X_{diff})$ ▷ Filtered attack magnitudes

$Su \leftarrow Segment(X))$

$Su_x \leftarrow Map(X_{difft}, Su)$ ▷ Group attack magnitudes based on segmentation

$Su_d \leftarrow Calculate\_density(Su_x)$ ▷ Calculate attacks per segment

return $Su(TopK\_indices(Su_d))$

3.1 White-box adversarial attacks

Adversary can attack a DNN by adding engineered noise to the input to increase the associated loss value, if it has some prior knowledge of the DNN including the weights and biases. AXAI utilizes Projected Gradient Descend (PGD) attack [11], although any ℓ_2 adversarial attack can replace PGD in our algorithm (Appendix B). However, PGD provides specific benefits such as stability and gradient smoothness that other attacks do not. PGD can be thought of as an iterative version of ℓ_2 Fast Gradient Method (FGM) attack [24], where in each iteration, the adversarial changes are clipped into an ℓ_2 ball of some ϵ value. PGD is generally considered a strong stable attack and is defined as,

$$x^{t+1} = \sqcap_{x+S}(x^t + \epsilon \nabla_x L(\Theta, x, y)), \tag{1}$$

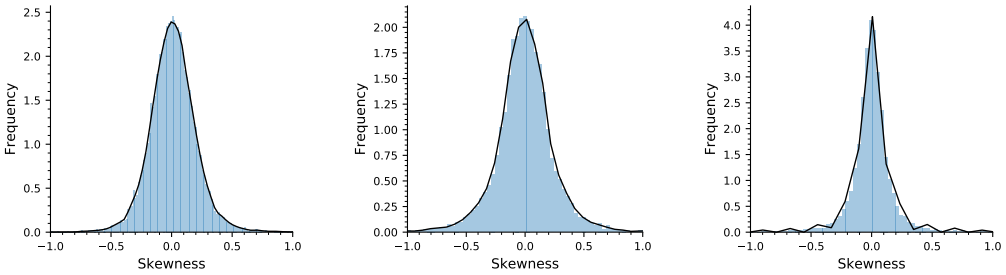
where for t iterations, x and y are the inputs and outputs, and Θ are the weights and biases.

3.2 Statistical analysis of attack magnitudes vs. frequency distributions

Here, we briefly report our statistical analysis of attack magnitudes vs. frequency distributions for a fixed DNN, dataset and an adversarial attack. We can show that the distributions are similar in their "shapes," "means," "mean ranks," "medians," and "quantiles," and follow a Beta distribution with specific parameters. Given that there is no significant difference in the distributions, we can provide a universal threshold using quantiles which separates the important features from the rest to produce explanations.

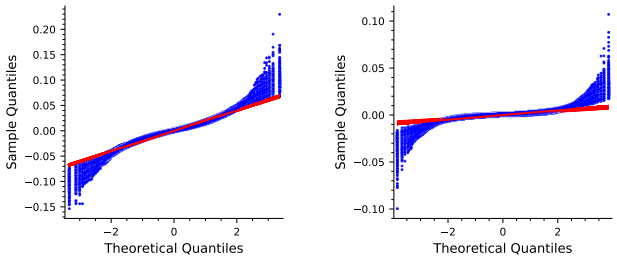
We can measure the symmetricity of distributions using the Fisher-Pearson coefficient of skewness. We present the results for AlexNet on CIFAR10 [25], VGG16 on CIFAR100 [26] and ResNet34 on ImageNet [27]. The Fisher-Pearson coefficients of the attack magnitudes vs. frequency distributions for all cases are shown in Fig. 2. It is seen that the skewness of all distributions falls within the $[-0.5, 0.5]$ range showing strong evidence that they are approximately symmetric [28]. Only 0.9% of CIFAR10, 3.3% of CIFAR100 and 1.9% of ImageNet test datasets lie outside of $[-0.5, 0.5]$ range.

Quantile-Quantile (Q-Q) plot allows us to understand how the quantiles of a distribution deviate from a specified theoretical distribution. The theoretical distribution selected is the normal distribution. The x-axis and y-axis represent the quantile values of the theoretical and sample distributions, respectively. While it is unlikely to have identical distributions that perfectly match, one can look at different parts of the Q-Q plot to distinguish between the similar and dissimilar locations in the distributions. Fig. 3 shows the Q-Q plots for random subsets of ImageNet and CIFAR10 test datasets each containing 1000 images. It is seen that the distributions follow a fairly straight line in the middle portion of the curve, while deviating at the upper and lower parts. This provides some evidence supporting the hypothesis that distributions follow a 'near-normal' distribution with heavier tails.



(a) PGD, AlexNet, CIFAR10 (b) PGD, VGG16, CIFAR100 (c) PGD, ResNet34, ImageNet

Figure 2: The Fisher-Pearson coefficient of attack magnitudes vs. frequency distributions.



(a) PGD, AlexNet, CIFAR10 (b) PGD, ResNet34, ImageNet

Figure 3: The Q-Q plot of sample distributions vs. theoretical normal distribution (mean=0, std=1).

	t-test (CIFAR10)	Mann-Whitney (CIFAR10)	t-test (ImageNet)	Mann-Whitney (ImageNet)
p-value	0.70	0.58	0.64	0.55

Table 1: p-values for the mean similarity statistical tests at significance level 0.05.

We perform the two-sample location t-test and Mann-Whitney U test to determine if there is a significant difference between two groups where the null hypothesis is the equality of the means. Carrying out pair t-tests on all samples allows us to be conservative in confirming the mean similarity of the distributions. A sample here is defined as the attack magnitudes vs. frequency distribution for a data point in the test adversarial dataset created by the PGD attack on a DNN trained on the training dataset. The results reported in Table 1 indicate no significant difference between the means. Further, the Mann-Whitney U test results indicate that all pairs are similar to each other on the mean ranks. Under the assumption of two distributions having similar shapes, one could further state that Mann-Whitney test can be considered as a test of medians [29]. Since, we have shown that the shapes are similar, we can conclude that there are no significant difference between the medians of the distributions. Further details in addition to the results for the ANOVA test are given in Appendix C.

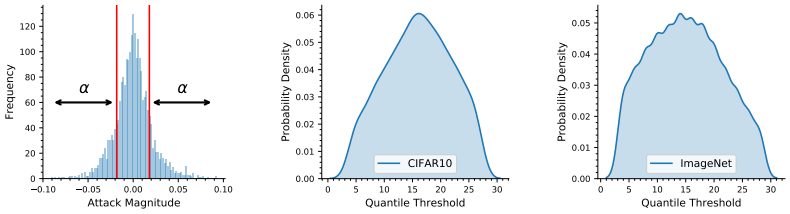
Next, to show consistency across distributions for a given model, dataset and attack, we estimate the values of quantiles, means and medians. We do this by estimating the statistics of the distributions and constructing confidences intervals. For each experiment, we estimate the mean, median, 15th, 25th, 75th and 85th quantiles of each attack magnitude vs. frequency distribution for the entire test dataset. The statistical confidence interval estimations at confidence level of 95% are reported in Table 2. Our results show that the confidence intervals have narrow ranges and the estimations are consistent. The estimates for the 15th, 25th, 75th and 85th quantiles indicate a strong symmetricity with respect to the origin in all cases. This matches the results of the skewness test in Fig. 2. Another observation is that the confidence interval of the mean and medians are pretty narrow, supporting the results of the t-tests and Mann-Whitney U test. Finally, we can show with high confidence that the distributions consistently follow a beta distribution. The beta distribution is a family of distributions defined by two positive shape parameters, denoted by p and q. The estimated p and q of the beta distribution are reported in Table 3. Further technical details on our analyses presented in this section, in addition to further experiments with audio and text input types, are provided in Appendix C.

	AlexNet, CIFAR10, PGD	VGG16, CIFAR100, PGD	ResNet34, ImageNet, PGD
15th Quantile	$(-1.807e-02, -1.805e-02)$	$(-1.419e-02, -1.414e-02)$	$(-1.785e-03, -1.777e-03)$
25th Quantile	$(-1.145e-02, -1.071e-02)$	$(-8.153e-03, -8.110e-03)$	$(-1.015e-03, -1.101e-03)$
Mean	$(1.775e-05, 2.295e-05)$	$(-6.850e-06, -3.624e-06)$	$(-1.090e-07, -6.000e-08)$
Median	$(2.115e-06, 1.127e-05)$	$(-2.842e-06, 4.467e-06)$	$(-2.155e-07, -9.381e-08)$
75th Quantile	$(1.071e-02, 1.073e-02)$	$(8.102e-03, 8.146e-03)$	$(1.011e-03, 1.016e-03)$
85th Quantile	$(1.809e-02, 1.812e-02)$	$(1.413e-02, 1.418e-02)$	$(1.777e-03, 1.785e-03)$

Table 2: Estimations for mean, median, 15th , 25th, 75th and 85th quantiles at 95% confidence level.

	AlexNet, CIFAR10, PGD	VGG16, CIFAR100, PGD	ResNet34, ImageNet, PGD
p	$(1.124e+01, 1.132e+01)$	$(2.129e+01, 2.171e+01)$	$(1.306e+02, 1.329e+02)$
q	$(1.136e+01, 1.145e+01)$	$(2.124e+01, 2.164e+01)$	$(1.303e+02, 1.326e+02)$

Table 3: Statistical estimations for parameters of beta distribution at 95% confidence level.



(a) (b) (c)

Figure 4: Visualization of the re-attacking process where only portions of inputs lying outside the red lines are attacked ($[0\%, \alpha\%]$, $[(100-\alpha)\%, 100\%]$) (b) AlexNet, CIFAR10 (c) ResNet34, ImageNet

	CIFAR10, AlexNet	ImageNet, ResNet34		CIFAR10, AlexNet	ImageNet, ResNet34
Attack Percentile			Attack Percentile		
$15\% - 85\%$	0.78	0.88	$0\% - 15\%\&85\% - 100\%$	0.16	0.07
$10\% - 90\%$	0.26	0.79	$0\% - 10\%\&90\% - 100\%$	0.26	0.13
$5\% - 95\%$	0.50	0.63	$0\% - 5\%\&95\% - 100\%$	0.45	0.25
$1\% - 99\%$	0.07	0.12	$0\% - 1\%\&99\% - 100\%$	0.92	0.80

Table 4: Adversarial test accuracy where only features within a certain percentile of the attack magnitudes vs. frequency distributions are attacked (PGD with 20 Iterations).

3.3 Quantile selection for the explanations

Our algorithm produces explanations that rely only on the features in the input that have the largest effect on the predictions. While the majority of the input is attacked, our belief is that only important features are strongly attacked. We show how one can select the boundary threshold between "explainable features" and the rest based on attack magnitudes. We demonstrate this with 2 experiments: 1) AlexNet trained on CIFAR10, 2) ResNet34 trained on ImageNet, both attacked by PGD with 20 iterations. In each case, we select the successfully attacked inputs from the adversarial test dataset, i.e., the inputs that fool the DNN. We then only re-attack specific features of the original clean inputs within the $[0\%, \alpha\%]$ and $[(100-\alpha)\%, 100\%]$ percentile of the distributions, where α is the percentage threshold. The re-attacking process starts from $\alpha = 0$, where none of the input features are attacked, and then we gradually increase the value of α until the attack successfully changes the prediction, and then we save the value of α (Fig. 4a). We repeat this for every input. The probability density distribution of α's are given in Fig. 4b and Fig. 4c with an estimated mean of $\alpha = 15$.

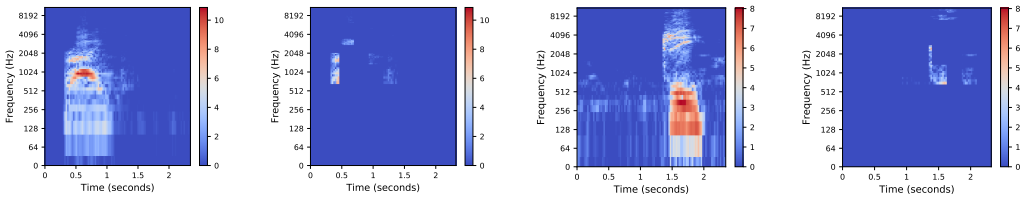
Further, we report the test accuracies of the DNNs on the adversarial test datasets that are created based on different attack percentiles. Given an attack percentile range, the adversarial test dataset consists of adversarial test inputs which are created by attacking only portions of the input features that lie withing a specific percentile range of the attack magnitudes vs. frequency distributions similar to above. This allows us to understand how the features lying in the middle area, tails and outliers of the distributions affect the DNN's predictions. Our findings are reported in Table 4. Our results show that the majority of the input features including those within the first two standard deviations and the outliers of the distributions do not have a strong effect on the predictions. A smaller portion of the input features which are also those attacked with the highest intensity, i.e., within the $[0\%, 15\%]$ and $[85\%, 100\%]$ percentiles of the distributions have the largest effect on the DNN's predictions, confirming our hypothesis. We see the same trend across different DNNs and datasets (Appendix C).

4 Experiment Results

Earlier, we provided a sample explanation created by AXAI for an image classifier. Appendix E contains more experiments for image classification and object detection DNNs. Further, Appendix E contains an ablation study and an interesting comparison between explanations produced by a non-robust model and an adversarially robust model. Here, we provide sample explanations produced by our algorithm for speech recognition and language-based tasks.

4.1 Explaining a speech recognition model

The Speech Commands Dataset [30] is an audio dataset of short spoken words. Here, we have converted the audio files to spectrograms and used them to train a LeNet model to identify "speech commands." We have created time-frequency segments by dividing the spectrogram into time-frequency grids similar to [31]. The x-axis and y-axis indicate the time-scale and log-scale frequency of the spectograms respectively, and the color bar indicates the magnitude. The spectrogram of the first word "Right" and its explanation are shown in Fig. 5a and Fig. 5b. The explanation shows that the first and last character in the spoken word "Right" stand out as important features ($[0.4s, 0.6s]$ and $[1.0s, 1.2s]$ intervals). This is reasonable because "Five" is the neighboring class of "Right" in the dataset (Appendix D) and "Right" and "Five" differ in the pronunciation of "r" and "f" and "t" and "v." The second example is for the word "Three" (Fig. 5c and Fig. 5d). The produced explanation indicates the importance of "Thr" ($[1.4s, 1.7s]$ interval). This is reasonable because "Three" and its neighbor "Tree" differ in the letter "h" in "Thr," and this difference is learned by the model during training to identify the two words correctly. More details on this experiment are given in Appendix E.



(a) "Right" (b) Explanation (c) "Three" (d) Explanation

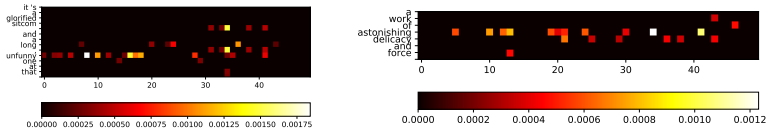
Figure 5: The AXAI explanations for the LeNet speech recognition model.

4.2 Explaining a text classification model

The Sentence Polarity Dataset [32] is a collection of movie-review documents labeled with respect to their overall sentiment polarity. Here, we will look at a negative and positive example (Fig. 6a and Fig. 6b) where the rows are the word tokens in the sentence, and the columns are the embedding dimensions. The NLP model used in our experiment is taken from [33] and trained on the dataset. As part of the pre-processing, the words in the dataset are tokenized and mapped to an embedding matrix. [34] mentions that the saliency map of an NLP model can be visualized using the embedding layer similar to saliency maps used for image-based models. Consequently, one can apply our algorithm to NLP models in a similar manner, i.e., we can utilize the first order derivative of the loss with respect to the word embedding. This technique is similar to what was used in [35]. The first example, "it's a glorified sitcom, and a long, unfunny one at that." is classified as a negative review by the model. Fig. 6a shows that the word "unfunny" is strongly highlighted as the main explanation for this prediction. For the positive example "a work of astonishing delicacy and force," it is seen that the word "astonishing" has the most significant influence on model's prediction.

4.3 Benchmark tests

We test our algorithm against LIME and SHAP (Gradient Explainer). It is important to note that SHAP subsumes a number of prior approaches and provides a fair baseline. To show the consistency of our approach, we present visualizations for 3 cases: 1) AlexNet, CIFAR10, 2) VGG16, CIFAR100, 3) ResNet34, ImageNet using the 3 explainability tools and provide more experiments in Appendix F. The algorithms produce similar explanations where AXAI has fewer tune-able parameters and



(a) Text example 1 (b) Text example 2

Figure 6: The AXAI explanations for the sentence classification model.

	Single CPU (Intel Core i5-7360U)	Single GPU (Tesla V100-SXM2)
LIME	105s	5.8s
SHAP (Gradient Explainer)	35s	3.8s
AXAI (PGD with 20 iters)	6.6s	1.7s

Table 5: Benchmark running-time experiments.

performs faster. LIME fails to produce good explanations for low-resolution CIFAR10 images. In Appendix F, we provide examples showing that AXAI outperforms LIME for low-resolution inputs. We benchmark the running-time performance of AXAI, LIME and SHAP for ResNet34 trained on ImageNet on a single CPU (Intel Core i5-7360U) and single GPU (Tesla V100-SXM2) on the entire test dataset. The results are given in Table. 5. LIME is the slowest to produce explanations. This is because LIME needs to forward propagate the perturbed inputs through the DNN several times. SHAP also slower to generate the results in comparison to AXAI. LIME works better on a GPU. AXAI maintains its relative performance on the CPU and GPU. This is because the segmentation step which mainly uses the CPU is the main computational bottleneck for the algorithms (Appendix A).

5 Final Remarks and Conclusion

In this paper, we proposed a new approach for explaining the predictions of DNNs. Interpretability is directly related to the readability of an explanation [36]. An explanation relying on thousands of features is not interpretable. AXAI, similar to LIME, uses input segmentation to create human-readable explanations focused on important input features. Further, AXAI has the following properties,

Property 1 (Robustness): Our approach is more robust to the changes in segmentation hyper-parameters in comparison to other segmentation based approaches such as LIME. This is because AXAI does not require a surrogate model trained on "randomly perturbed inputs." AXAI uses the deterministic attack magnitudes as "base explanations" for a given DNN and dataset, and uses segments as an "aid" to visualize the results. The segmentation affects the visualizations. We further explain this in Appendix A. Robustness is identical to stability of explanations as defined in [37]. A lower number of non-deterministic steps in the algorithm enhances stability. A carefully filtered explanation based on our approach simply removes the features that have a low impact on predictions. One can interpret this process as a de-noising step to create a sparse representation of explanations.

Property 2 (Local attribution): Our algorithm is locally stable and uses local attributes to produce explanations. This is because an adversarial attack uses the most minimal amount of noise within an ℓ_2 ball of some small ϵ to fool the DNN. Given the un-targeted nature of the attack used in AXAI, the distributions can be interpreted as estimations of the boundaries among neighboring classes. Thus, one can conclude that the attack magnitudes are a representation of feature contributions to the predictions on a local scale. A similar conclusion is made in [38], where it is argued that gradients can in fact point to important local attributions of a DNN. We explore this in details in Appendix D.

Property 3 (Completeness): Completeness as a property is described as the ability to accurately explain the operations of a DNN [36]. An explanation is more complete when it can explain the behavior of the DNN for a larger set of inputs. [14] and [13] mention the problem of sensitivity and lack of stability in gradient-based algorithms. In the literature, if a solution can reduce the gradient "sensitivity" problem, it can be described as having the "completeness" property [36]. AXAI with PDG attack is complete in the same sense as SmoothGrad is [13]. SmoothGrad takes the average of saliency maps with added Gaussian noise to reduce sensitivity. The PGD attack behaves in a similar manner by adding adversarial noise at each iteration. Both solutions add perturbations to the input to smooth gradient fluctuations. While further research can be done on the power of iterative attacks in their gradient smoothing effects, we argue that AXAI with iterative PGD does have the desirable characteristic and produces stable sharpen visualizations of sensitivity maps for robust explanations.

Broader Impact

Our work in this paper contributes to the fields of adversarial machine learning and artificial intelligence (AI) explainability. There is still a huge gap between building a model in Jupyter notebook and shipping it as a stand-alone product to the users. Advances in these two fields directly relate to deploying AI systems that behave in a robust and user-friendly manner after deployment. Building AI systems is hard. AI explainability can provide insights into how AI models behave, why they make the decision they make and the reasoning behind their incorrect predictions. Additionally, explaining the outcomes of a model can help reduce bias and contribute to improvements in accountability and ethics by providing beneficial insights into how AI models think and make their decisions.

Despite the hype, AI engineers struggle with deploying models which meet the users' performance expectations. A lack of robustness in the performance of trained model is a major impediment. We need to be able to design AI systems that both perform well and are robust. A robust model not only makes correct predictions in expected environment, but it also maintains an acceptable level of performance in unpredictable situations. Our work gives insights into how the adversary attacks an AI system trained to perform a specific task. Understanding how adversarial attacks behave can help AI engineers in development of AI systems that perform as expected while maintaining some level of robustness in presence of external disturbances and adversarial noise. This type of information can help AI engineers in developing AI models that perform better. In short our paper can help AI researchers in their endeavor to design, develop and deploy explainable ethical AI systems that are robust and reliable.

References

[1] Gil Fidel, Ron Bitton, and Asaf Shabtai. When Explainability Meets Adversarial Learning: Detecting Adversarial Examples using SHAP Signatures. *arXiv preprint arXiv:1909.03418*, 2019.

[2] Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41(3):647–665, 2014.

[3] Anupam Datta, Shayak Sen, and Yair Zick. Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In *IEEE Symposium on Security and Privacy (SP)*, pages 598–617. IEEE, 2016.

[4] Sina Mohseni, Niloofar Zarei, and Eric D Ragan. A survey of evaluation methods and measures for interpretable machine learning. *arXiv preprint arXiv:1811.11839*, 2018.

[5] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

[6] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[7] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[8] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-Cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

[9] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, 2016.

[10] David Alvarez-Melis and Tommi S Jaakkola. A causal framework for explaining the predictions of black-box sequence-to-sequence models. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.

[11] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[12] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the International Conference on Machine Learning*, pages 3145–3153. JMLR. Org., 2017.

[13] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: Removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[14] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the International Conference on Machine Learning*, pages 3319–3328. JMLR. Org., 2017.

[15] Alon Jacovi, Oren Sar Shalom, and Yoav Goldberg. Understanding convolutional neural networks for text classification. *the EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, 2018.

[16] Guannan Zhao, Bo Zhou, Kaiwen Wang, Rui Jiang, and Min Xu. Respond-Cam: Analyzing deep models for 3D imaging data by visualizations. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 485–492. Springer, 2018.

[17] 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), 2015.

[18] Sören Becker, Marcel Ackermann, Sebastian Lapuschkin, Klaus-Robert Müller, and Wojciech Samek. Interpreting and explaining deep neural networks for classification of audio signals. *arXiv preprint arXiv:1807.03418*, 2018.

[19] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

[20] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics*, 9(3):1350–1371, 2015.

[21] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, pages 125–136, 2019.

[22] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. *International Conference on Learning Representations (ICLR)*, 2019.

[23] Andrea Vedaldi and Stefano Soatto. Quickshift and kernel methods for mode seeking. In *European Conference on Computer Vision*, pages 705–718. Springer, 2008.

[24] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *International Conference on Learning Representations (ICLR)*, 2015.

[25] P Kaur. Convolutional neural networks (CNN) for cifar-10 dataset." parneetk. github. io/blog/cnn-cifar10/, 2017, 2018.

[26] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6, 2009.

[27] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE, 2009.

[28] Michael George Bulmer. *Principles of Statistics*. Courier Corporation, 1979.

[29] John H McDonald. *Handbook of Biological Statistics*, volume 2. Sparky House Publishing Baltimore, MD, 2009.

[30] Pete Warden. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition, 2018.

[31] Saumitra Mishra, Bob L. Sturm, and Simon Dixon. Local interpretable model-agnostic explanations for music content analysis. In *ISMIR*, 2017.

[32] Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

[33] Yoon Kim. Convolutional Neural Networks for Sentence Classification, 2014.

[34] Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and Understanding Neural Models in NLP, 2015.

[35] Takeru Miyato, Andrew M. Dai, and Ian Goodfellow. Adversarial Training Methods for Semi-Supervised Text Classification, 2016.

[36] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining Explanations: An Overview of Interpretability of Machine Learning, 2018.

[37] Marko Robnik-Šikonja and Marko Bohanec. *Perturbation-Based Explanations of Prediction Models*, pages 159–175. 2018.

[38] Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

[39] Adam Van Etten, Dave Lindenbaum, and Todd M Bacastow. SpaceNet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*, 2018.

A QuickShift Segmentation

QuickShift is a mode seeking clustering algorithm proposed by [23]. QuickShift creates segments by repeatedly moving each data point to its closest neighbor point that has higher density calculated by a Parzen Estimator. The Kernel size argument in the QuickShift function controls the width of the gaussian kernel of the estimator. The path of moving points can be seen as a tree that connects data points. Eventually, the algorithm connects all data points into a single tree. To balance between under and over fragmentation of the image, a threshold, τ, is served as a breaking point that limits the length of the branches in the QuickShift trees. The threshold, τ, is the Max distance argument in the QuickShift function. Finally, the pre-processing step of QuickShift projects a given image into a 5D space, including color space (r, g, b) and location (x, y). A hyper-parameter, λ, takes a value between 0 and 1 and serves as a weight assigned to the color space, such that the feature space can be presented as $\{\lambda r, \lambda g, \lambda b, \lambda x, \lambda y\}$.

LIME uses QuickShift for image segmentation where the default Kernel Size is 4, the Max distance is 200, and the threshold τ is 0.2. This combination prevents generating too many image segments. Even-though the image segmentation process is only performed once per image, we would like to point out that the parameter selection does change the explanation results slightly. First, increasing the kernel size increases the computation time while decreasing the number of image segments, making this parameter the major computational bottleneck in image segmentation. Second, extra care should be taken when it comes to low-resolution images, when the image is coarse and the number of image segments are low, because important and unimportant features can easily be merged together, as demonstrated in Fig. 7. From the perspective of explainability, both accuracy and human-readability are needed. This is achieved as long as the important segments are not merged with unimportant ones. This problem can be solved by selecting a small kernel size. In our algorithm, we introduce a user tunable hyper-parameter, called explainabilty length, K, that allows users to decide the number of explainable segments. Human-readability is subjective, so we let the user decide the explainable length, Fig. 8. We see that in Fig. 8, the wall of the castle on the left most side of the image is merged with the sky due to the similarity between colors. In both case, we picked the top 10 segments as explanations, i.e., explainabilty length=10. It is important to note that unlike LIME and other explainability algorithms, the choice of a longer explainabilty length (more segments) does not increase the computational time of our algorithm.

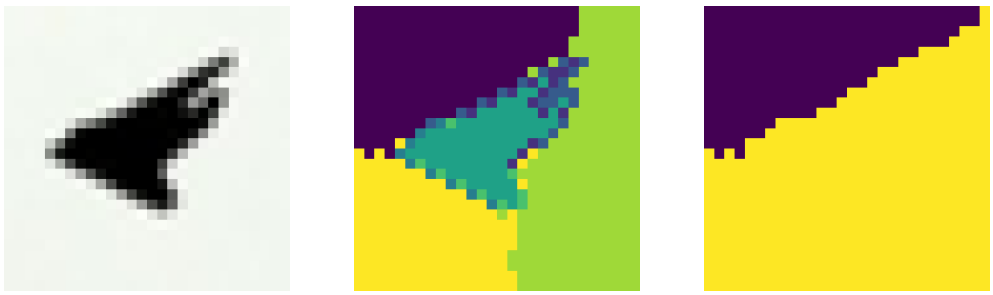
(a) Original image from CIFAR10 (b) Preferable segmentation (c) Under segmentation

Figure 7: Segmentation in low-resolution images.

Deciding the tradeoff between the importance of the color (r,g,b) and spatial components (x,y) of the feature space, is especially important for high resolution images. Take a castle image in the ImgeNet dataset as an example (given in Fig. 9). We choose two different parameter combinations for comparison. The only difference between the two combinations is the λ parameter. For the first combination, we used 0.2 (Fig. 9b), for the second combination, we used 0.8 (Fig. 9c). One can see that using a lower λ prevents details from merging with irrelevant background information. In Fig. 9b and Fig. 9c, the total number of segments are nearly the same (73 and 81) but the explanations have different qualities.

(a) Original image from ImageNet (b) Explainabilty length = 1 (c) Explainabilty length = 2

Figure 8: Relationship between produced explanations and explainabilty length.



(a) Original image from ImageNet (b) λ=0.2 (c) λ=0.8

Figure 9: The effects of λ on the explanations produced.

B Convergence of Explanations across Adversarial Attacks

As a tool for explainability, efficiency, accuracy and consistency are of top priority. Our experiments show that ℓ_2 PGD attacks with different iterations create explanations similar to ℓ_2 FGM attack. This points to consistency in explanations produced by our algorithm. PGD attack is an iterative version of FGM, while both attacks are subjected to an ℓ_2 norm. Note that the distribution of the attacks can influence the explanation results. This also means that since the attack distributions of the first iteration and later iterations of the PGD attack are nearly identical, the overall explanations remain the same. In Fig. 10, we provide an example from the ImageNet dataset to show the convergence of the attacks and consistency of our explanations. Fig. 10b shows the explanation results for an FGM based algorithm. Fig. 10c and Fig. 10d show the explanation results based on the PGD attack with different number of iterations. They both look exactly the same. This is because the slight changes on the attack distribution for different number of iterations, do not affect the overall density of pixel changes in each segment, thus the final explainability results do not change. This point to stability and consistency of our algorithm. To further explore the stability and consistency of our approach, we can segment the image into much smaller segments, as given in Fig. 10e and Fig. 10f, in this case using 50 times more segments than the previous case and then produce the explanations. In this case, we do see small differences between an explanation produced with a PGM attack with 10 iterations and one based on a PGM attack with 40 iterations. These small differences are caused by small differences in the attack distributions in each segment. While it is interesting to further explore how different types of attacks can lead to more "suitable" explanations, it is important to note that one could explain the outcomes using our algorithm and with both types of attacks. Further, we can conclude that using FGM or PGD attacks in our algorithm satisfies consistency, accuracy and efficiency conditions for producing explanations.

(a) Original Image from ImageNet (b) ℓ_2 FGM (c) PGD with 10 iterations

(d) PGD with 40 iterations (e) PGD with 10 iterations (f) PGD with 40 iterations

Figure 10: Convergence of explanations for different adversarial attacks and number of segments (Architecture: ResNet34, Dataset: ImageNet).

C Further Details on the Statistical Analyses given in Subsection 3.2

C.1 Further details on the statistical tests

The Fisher-Pearson coefficient g_1 of a distribution x with a sample size N is calculated using the third moment m_3 and the second moment m_2 of the distribution,

$$g_1 = \frac{m_3}{m_2^{\frac{3}{2}}}, \tag{2}$$

where,

$$m_i = \frac{1}{N}\sum_{n=1}^{N}(x[n] - \bar{x})^i \tag{3}$$

If skewness is 0, the data is perfectly symmetrical, if skewness is positive, then one interprets the distribution as skewed right, if skewness is negative, then the distribution is skewed left. [28] pointed out that there are three levels of symmetricity, a) when skewness is between -0.5 to 0.5, the distribution is "approximately symmetric," b) when skewness is within -1 and -0.5 or 0.5 and +1, the distribution is "moderately skewed," c) when skewness falls out of the mentioned range, then the distribution is highly skewed. The Fisher-Pearson coefficient of all attack magnitudes are shown in Fig. 15. It is seen that the skewness of all attack magnitudes falls within -0.5 an 0.5 showing the strong evidence that the distributions are approximately symmetric.

The t-statistic test is represented as follows,

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_p\sqrt{\frac{2}{n}}} \tag{4}$$

where,

$$s_p = \sqrt{\frac{s_{X_1}^2 + s_{X_2}^2}{2}} \tag{5}$$

Here $\bar{X_1}$, $\bar{X_2}$ and $s^2_{X_1}$, $s^2_{X_2}$ are the means and variances of the two distributions with size n. The t-statistic can be interpreted as a kind of measurement for the ratio of the "difference between groups" over the "difference within groups." Carrying out pair t-tests on all samples allows us to further be conservative on the similarity on means between the distributions. The results are shown in Table 4. Overall, there is no significant differences between the distributions.

To show the similarity between the distributions produced for a dataset, we also use the one-way ANOVA test on all the samples to show that the means across different distributions are the same. Samples here are defined as intensity vs. frequency distributions for all adversarial test samples created by attacking a model trained on a specific dataset. For CIFAR10, we get the p-value of 0.9, and for a random subset of ImageNet test dataset we get the p-value of 0.94, indicating no significant differences between the distribution means. Similarly, a two-sample location t-test is used to determine if there is a significant difference between two groups where the null hypothesis is the equality of the means. Even-though ANOVA and t-tests are known for being robust on non-normal data, we further performed pair wise Mann–Whitney U test on all pair of distributions to test whether the mean ranks are similar.

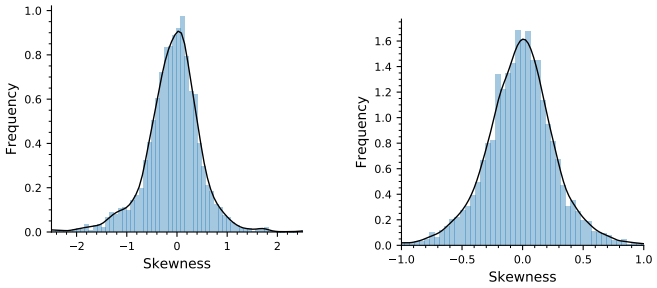
Mann–Whitney U test is a nonparametric test of the null hypothesis that two independent samples selected from population have the same distribution. The statistic U is calculated as following,

$$U_1 = R_1 - \frac{n_1(n_1-1)}{2},\ U_2 = R_2 - \frac{n_2(n_2)}{2} \tag{6}$$

Where subscripts "1" and "2" denote the two distributions being compared. In the case of comparing two distributions "sample 1" and "sample 2." One first combines "sample 1" and "sample 2" together to form an ordered set, and then one assigns ranks to the members of this set. Next, one adds up the ranks for the members of the set coming from "sample 1" and "sample 2" respectively. This is called the rank sum of R_1 and R_2. Once the rank sums are calculated The U statistic of the two distributions (U_1 and U_2) are calculated as above. Finally, the U statistic is determined by the lower value between U_1 and U_2. If U_1 is lower than U_2, then U_1 is the U statistic of the Mann Whitney test between "sample1" and "sample 2"" and vice versa. We further perform the pair-wise Mann–Whitney U test on all pair of distributions to test whether the mean ranks are similar as well. If U is 0, it means that the two distributions are far away from each other where there are no overlaps between them. If the Rank sums are close enough, one can say the two distributions are highly overlapped. Thus, one can say the Mann–Whitney U test is a test comparing the Rank sums (or the mean ranks, calculated by dividing the Rank sums over the size of samples) of two distributions. The smaller values of U_1 and U_2 is the one used when consulting significance tables.

C.2 Quantile-Quantile plot

Quantile-Quantile (Q-Q) plot allows us to show how the quantiles of a distribution deviates from a specified theoretical distribution. The theoretical distribution selected here is the normal distribution. Quantiles are cut points dividing the range of a probability distribution into continuous intervals with equal probabilities. A Q-Q plot is then a scatter-plot showing two sets of quantiles (a sample distribution and a theoretical distribution) against one another. The x-axis are the quantile values of the theoretical distribution while the y-axis are the quantile values of the sample distribution, i.e., the distribution of attack intensities vs. pixel frequencies. One can see that if the quantiles of the sample distribution perfectly match the theoretical quantiles, then one can see all the quantiles located on a straight line. While it is unlikely to have identical distributions that perfectly match the theoretical distribution, one can look at different sections of the Q-Q curves to distinguish the parts that two distributions share similarity and parts that they differ. Compared to a normal distribution, if the sample distribution has heavy or light tails, the Q-Q curve bends at the upper or lower portion based on side of the tails that deviates from the normal distribution. One can say that one purpose of Q-Q plots is to look at the "straightness" of the Q-Q curve. We took a subset that contains 1000 images from both ImageNet and CIFAR10 and plotted the distributions against a normal distribution as given in Fig. 3. It is seen that all attack distributions plotted against the normal distribution have fairly straight lines at the middle portion of the Q-Q curve, while the curve bends at the upper part and the lower part. One can interpret this result as the attack magnitudes are similar to a normal distribution but differ in a way that the distributions have"heavy tails" thus the upper part of the curve bends "up" and the lower part of the curve bends"down."



(a) PGD, LeNet, SpeechCommands

(b) PGD, CNN, Sentence Polarity

Figure 11: The Fisher-Pearson coefficient of attack magnitudes vs. frequency distributions.

C.3 The beta distribution

The beta distribution is a family of distributions defined on the interval [a, b] parametrized by two positive shape parameters, denoted by p and q. The general formula for the probability density function of the beta distribution can be written as,

$$f(x) = \frac{(x-a)^{p-1}(b-x)^{q-1}}{B(p,q)(b-x)^{p+q-1}} \tag{7}$$

where,

$$B(\alpha,\beta) = \int_0^1 t^{\alpha-1}(1-t)^{\beta-1}dt \tag{8}$$

The beta distribution is often used to describe different types of data, such as rainfall, traffic and financial data. In this paper, estimate the parameters of a beta distribution for our distributions. The method of moments estimation is employed to calculate the shape parameters, p,q, of the two-parameter beta distribution. As the interval [a, b] is known, the method of moments estimates of p and q are

$$p = \bar{x}(\frac{\bar{x}(1-\bar{x})}{s^2} - 1) \tag{9}$$

$$q = (1-\bar{x})(\frac{\bar{x}(1-\bar{x})}{s^2} - 1) \tag{10}$$

When the interval [a, b] is [0, 1]. This is called the standard beta distribution. Since in most cases the interval [a, b] is not bounded between [0, 1], one can replace $\bar{x}$ with $\frac{\bar{x}-a}{b-a}$ and s^2 with $\frac{s^2}{(b-a)^2}$. Finally the estimated p and q of the beta distribution is listed in Table 3.

C.4 Statistical analysis of distributions for DNNs with text or audio input types

We test the symmetricity of distributions by calculating the Fisher-Pearson coefficient of skewness for LeNet trained on Speech Commands dataset, and a convolutional neural network (CNN) given in [33] on Polarity dataset. The Fisher-Pearson coefficients of the attack magnitudes vs. frequency distributions for all 3 cases are shown in Fig. 11. It is seen that the skewness of all distributions falls within the $[-0.5, 0.5]$ range showing strong evidence that they are approximately symmetric [28].

We perform the two-sample location t-test and Mann-Whitney U test to determine if there is a significant difference between two groups where the null hypothesis is the equality of the means. The results reported in Table 6 indicate no significant difference between the means. Further, the Mann-Whitney U test results indicate that all pairs are similar to each other on the mean ranks. Under the assumption of two distributions having similar shapes, one could further state that Mann-Whitney test can be considered as a test of medians [29]. Since, we have shown that the shapes are similar, we can conclude that there are no significant difference between the medians of the distributions.

Next, to show consistency across distributions for a given model, dataset and attack, we estimate the values of quantiles, means and medians. We do this by estimating the statistics of the distributions

Dataset	LeNet, SpeechCommands, PGD		CNN, Sentence Polarity, PGD	
Test	t-test	Mann-Whitney	t-test	Mann-Whitne
p-value	0.30	0.25	0.47	0.42

Table 6: p-values for the mean similarity statistical tests at significance level 0.05.

	LeNet, SpeechCommands, PGD	CNN, Sentence Polarity, PGD
15th Quantile	$(-4.110e-3, -4.049e-3)$	$(-2.753e-1, -2.673e-1)$
25th Quantile	$(-1.150e-3, -1.109e-3)$	$(-1.472e-1, -1.414e-1)$
Mean	$(1.749e-5, 2.245e-5)$	$(-4.165e-3, -2.492e-3)$
Median	$(-4.181e-09, 1.356e-09)$	$(-2.142e-3, -6.219e-4)$
75th Quantile	$(1.145e-3, 1.204e-3)$	$(1.365e-1, 1.421e-1)$
85th Quantile	$(4.153e-3, 4.220e-3)$	$(2.599e-1, 2.677e-1)$

Table 7: Estimations for mean, median, 15th , 25th, 75th and 85th quantiles at 95% confidence level.

and constructing confidences intervals. For each experiment, we estimate the mean, median, 15th, 25th, 75th and 85th quantiles of each attack magnitude vs. frequency distribution for the entire test dataset. The statistical confidence interval estimations at confidence level of 95% are reported in Table 7. Our results show that the confidence intervals have narrow ranges and the estimations are consistent. The estimates for the 15th, 25th, 75th and 85th quantiles indicate a strong symmetricity with respect to the origin in all cases. Another observation is that the confidence interval of the mean and medians are pretty narrow, supporting the results of the t-tests and Mann-Whitney U test. Finally, we can show with high confidence that the distributions consistently follow a beta distribution. The beta distribution is a family of distributions defined by two positive shape parameters, denoted by p and q. The estimated p and q of the beta distribution are reported in Table 8.

D Explanations and Class Boundaries

Explaining how important features affect the predictions made by the model depends on the set of classes the model was trained to predict. Un-targeted attacks change the prediction label of an input to the label of its closest neighbor. Based on the different datasets that a model may have been trained on, the label changes after attack may be significantly different. For example, given an image of a "Beagle" and a model that is trained on a dataset consisting of labels {Cats and Dogs}, after attacking the model, the label of the image can change from "Dog" to "Cat." But if the same model is trained on a dataset composing of "Beagle, Golden retriever, and Egyptian Cat", the label of the image can change from "Beagle" to "Golden retriever," which is a more granule change. When an image is attacked, the features of the image will be directed to the nearest class with a similar probability distribution in the decision layer. Let's look at an example from ImageNet where the input image is classified as a "convertible" by ResNet34 trained on ImageNet (given in Fig. 12). There are multiple classes such as minivan, sports car, race car etc., under the "car" category in ImageNet. After attacking the model, the label changes from "convertible" to "sports car." This indicates that "sports car" may be the nearest neighbor class to the "convertible" class. If we look at the produced explanations we see that segments including the door are intensely attacked as given in Fig. 12b. The fact is that the model thinks that the doors are the 'most' important features for switching the label from "convertible" to "sports car." Both classes, "convertible" and "sports car," have similar wheels but different doors. In order to fool the model, attacking the wheels is not of top priority, it's the doors that makes the difference between two classes. The fact is that the model thinks that the doors are the most important features for classifying the original image as "convertible" and not "sports car." Both classes, have similar wheels but different doors. In order to fool the model, attacking the wheels is not of top priority, it's the doors that make the difference between two classes. After bluring the segments of interest to the model, i.e. the door segment—Fig. 12c, and feeding the image to the model, the predicted label changes from "convertible" to "sports car" which proves that the doors are the major features supporting the predictions made by the model. Using adversarial attacks as the

	LeNet, SpeechCommands, PGD	CNN, Sentence Polarity, PGD
p	$(5.282e+1, 5.451e+1)$	$(1.322e+1, 1.368e+1)$
q	$(5.144e+1, 5.309e+1)$	$(1.346e+1, 1.393e+1)$

Table 8: Statistical estimations for parameters of beta distribution at 95% confidence level.

force behind producing the explanations helps with finding the important features that are not only globally important to the model (doors are important features of cars, other classes do not have doors similar to cars), but also locally important to the model (within the car class, doors are the important features that make a difference between a convertible and a sports car).

(a) (b) (c)

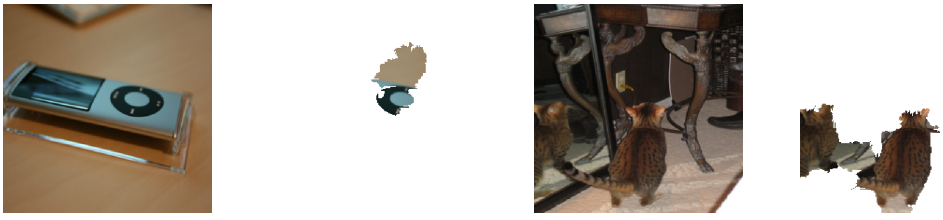
Figure 12: Left: Original image sample from ImageNet, Middle: The intensely attacked segments. Right: The original image with the explainable parts, i.e., the doors, blurred.

There are also some explainable features that humans hardly understand but models do, these can be called “non-robust features." [22] introduced the concept of robust and non-robust features, where the authors indicated that there are features that humans ignore but the models are sensitive to. They call these the non-robust features. Non-robust features are the features can easily be manipulated by the attacker in order to fool the model. Robust features are features that are both important to the model and also humans and at the same time invincible to small adversarial manipulations.

E Further Experiment Results

E.1 Explaining an image classification model

Fig. 13 shows two examples of the explanations produced using AXAI for image samples from ImageNet [27] test dataset for a Resnet34 trained on ImageNet training dataset. In the first example, Fig. 13a, the explanation results clearly show that the round control panel on an iPod is an important feature that helps the model identify an IPod in the image. The second example, Fig. 13c, shows how the model recognizes that there are two cats in the image (one is the reflection of the cat in the mirror).



(a) An image of an iPod (b) Explanation (c) An image of a cat (d) Explanation

Figure 13: The explanation results for a ResNet34 image classification model trained on ImageNet.

CIFAR10 dataset [25] consists of images of size 32×32 pixels, compared to ImageNet, these images are low-resolution images. Fig. 14 shows the explanations produced by AXAI for sample images from CIFAR10 dataset for an AlexNet image classification model trained on CIFAR10 training dataset. For CIFAR10, our explanations clearly separate the background and capture the target object. The explanation given in Fig. 14b shows that the head of the horse with the leather halter is

recognized by the model, and the white fence behind the horse is completely ignored by the model. This indicates that the model is well-trained. Similarly in Fig. 14d the ear and head of deer in the image helps the model to classify the image correctly into the deer class. Images from CIFAR10 dataset are easily explained due to the nature of the dataset with most objects in the images being located in the middle of the image and the lack of noisy background in most images.

(a) An image of a horse (b) Explanation (c) An image of a deer (d) Explanation

Figure 14: The explanation results for an AlexNet image classification model trained on CIFAR10.

E.2 Explaining an object detection model

We present two examples of explanations produced by our algorithm for a YOLOv3 object detection model trained on the SpaceNet Building Dataset [39] to detect buildings in overhead imagery. The produced explanation are clearly focused on areas where buildings are located and ignore empty spaces in the images such as the top left corner of Fig. 15b. Further, as seen in Fig. 15d, the roads are ignored and only buildings and their contours affect the predictions made by the object detector.

(a) Image sample 1 (b) Explanation (c) Image sample 2 (d) Explanation

Figure 15: The explanation results for a YOLOv3 object detection model trained on SpaceNet Building Dataset .

E.3 Further details on the speech recognition experiment

The Speech Commands Dataset [30] is an audio dataset of short spoken words, such as “Right,” “Three,” “Bed,” etc. The audio files are converted to spectograms and are used to train a LeNet for a command recognition task. A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. Fig. 5a is an example of a spectrogram. The y-axis, the frequency, of the spectograms are presented on a log-scale, the x-axis represent the time-scale, and the color bar shows the magnitude. Fig. 5a is the frequency spectrum of a human speaking the word “Right.” It is seen that in the time interval 0.4s to 1.1s, high magnitude is presented in the spectrum. In other words, the speaker pronounces the word "Right" around 0.4s to 1.1s into the recorded audio file. This is how one reads a spectrogram. Our explainable solution uses audio files as input, converts them into spectrograms, and then generates the corresponding explanations. So if one feeds AXAI with an audio file of a human speaking “Right,” AXAI first transforms the audio into a spectrogram shown in Fig. 5a, and produces the explanations in Fig. 5b. The explanation will have the exact same scale as the input, and simply masks out the unimportant parts of the spectrogram. To read the explanations, one can refer to the original spectrogram input Fig. 5a and find where the audio is located in the spectrogram (for example looking at the magnitudes), and then look at the corresponding location of the explanations in Fig. 5b.

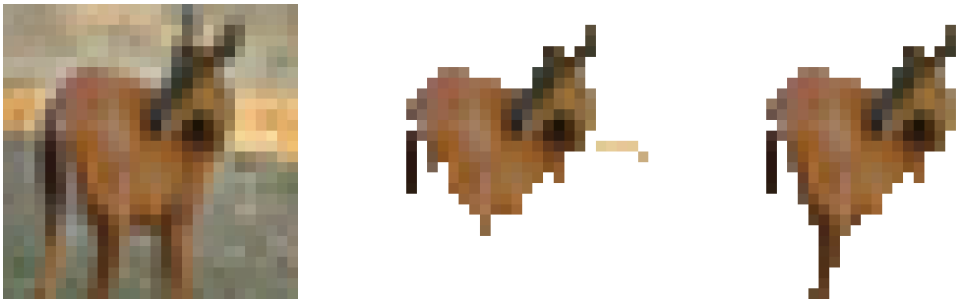
The explanations of two examples are presented in Fig. 5. The spectrogram of the first example "Right" and its explanation are shown in Fig. 5a and Fig. 5b. One can see from Fig. 5a that the spoken word "'Right" appears between 0.4s to 1.1s in the spectrogram of the audio file. If one looks at its corresponding explanation, it is seen that only time-intervals of 0.4s to 0.5s, 0.5s to 0.6s and 1.0s to 1.2s are not masked out by AXAI. This means that these intervals in the audio have great importance for the prediction made by the model. if we look back at Fig. 5a, one then realizes that the explanation shows that the first few and the last few seconds of the spoken word "Right" are important to the model, and the middle part is not. Why is that? The neighboring class of "Right" is "Five." "Right" and "Five" differ in how "R" & "F" and "t" & "ve" are pronounced. The middle part of "Five" and "Right" is highly similar and does not affect the model's prediction on deciding whether the spoken word is "Five" or "Right." The second example is "Three." As seen in the spectrogram, Fig. 5c, "Three" is expressed around the time-interval 1.4s to 2.2s in the spectrogram of the audio file. The corresponding explanation is shown in Fig. 5d. The explanation masks out almost everywhere except 1.4s to 1.6s and a small part in 1.6s to 1.7s and 1.9s to 2.2s. Now, let's look at the original spectrgram of "Three" and understand what the explanation means. Since The explanation highlights 1.4s to 1.6s, which is the first few seconds of the spoken word. To understand why, one can learn that if we attack the model, then "Three" is miss-classified as "Tree." This indicates that the model has learned to recognize "Three" and not "Three" by learning the difference between "Thr" and "Tr." The explanation tells us that the first few seconds of the audio are important (the utterance of "Thr").

E.4 Ablation study

If a feature or a group of features is important to a model, then completely removing those features from the input would decrease the probability of a correct prediction. Accordingly, we performed an ablation study confirming that the explanations produced by AXAI contain important features. This ablation method can be used to test the accuracy of an explainability solution. If the generated explanation is faithful to the model, then removing the explanations would decrease the accuracy of the predictions. In this section, we demonstrate a simple experiment to validate our algorithm. Our experiment is performed as follows: 1) Generate the explanation of a targeted image X via AXAI, where the explanation length $K = 10$ is selected in this experiment, 2) Blur the top 5 explanations/segments of the targeted image according to the produced explanations, feed the modified image to the model and obtain its label, 3) repeat this process throughout the test dataset 4) Calculate the total decrease in accuracy. We use a ResNet34 training on ImageNet for this experiment and report the results for the entire ImageNet test dataset. Our results show that the prediction accuracy of the DNN decreases to $\%43$ after blurring the top 5 explanation/segments. To further investigate, instead of blurring the top 5 explanations, we blur only the 6th to10th explanations. This results in a $\%22$ drop in total accuracy. Hence, we can conclude 1) AXAI generates faithful explanations so that blurring the top explanations (the 1st-5th explanations) lead to a strong decrease in model prediction accuracy, and 2) AXAI generates faithful explanations in order of importance, i.e., the generated 6th to 10th explanations are also important to the model but their influence on model predictions is relatively less than the first 5 generated explanations.

E.5 AXAI explanations for a robust model trained with adversarial training

In this subsection, we compare the explanations produced for a robust model to explanations produced for a non-robust model. In our experiment, a robust model is a model trained on an adversarial dataset in addition to the training dataset so that the final trained model is more robust against adversarial attacks. Hypothetically, a robust model should focus more on robust important input features when making predictions. We have trained a non-robust AlexNet and a robust AlexNet on CIFAR10 and produced the explanations using AXAI for test inputs. Fig. 16 shows the AXAI produced explanations for the DNN given a sample input. It is seen that a small part of the background is included in the explanations produced for the non-robust AlexNet. However, the AXAI generated explanations for the robust model includes only the important features pertaining to the object in the image. In addition, the leg of the deer is now included in the explanations as well. It is concluded that explanations produced for the robust DNN are sharper, clearer and more robust than the ones generated for the regularly trained DNN.

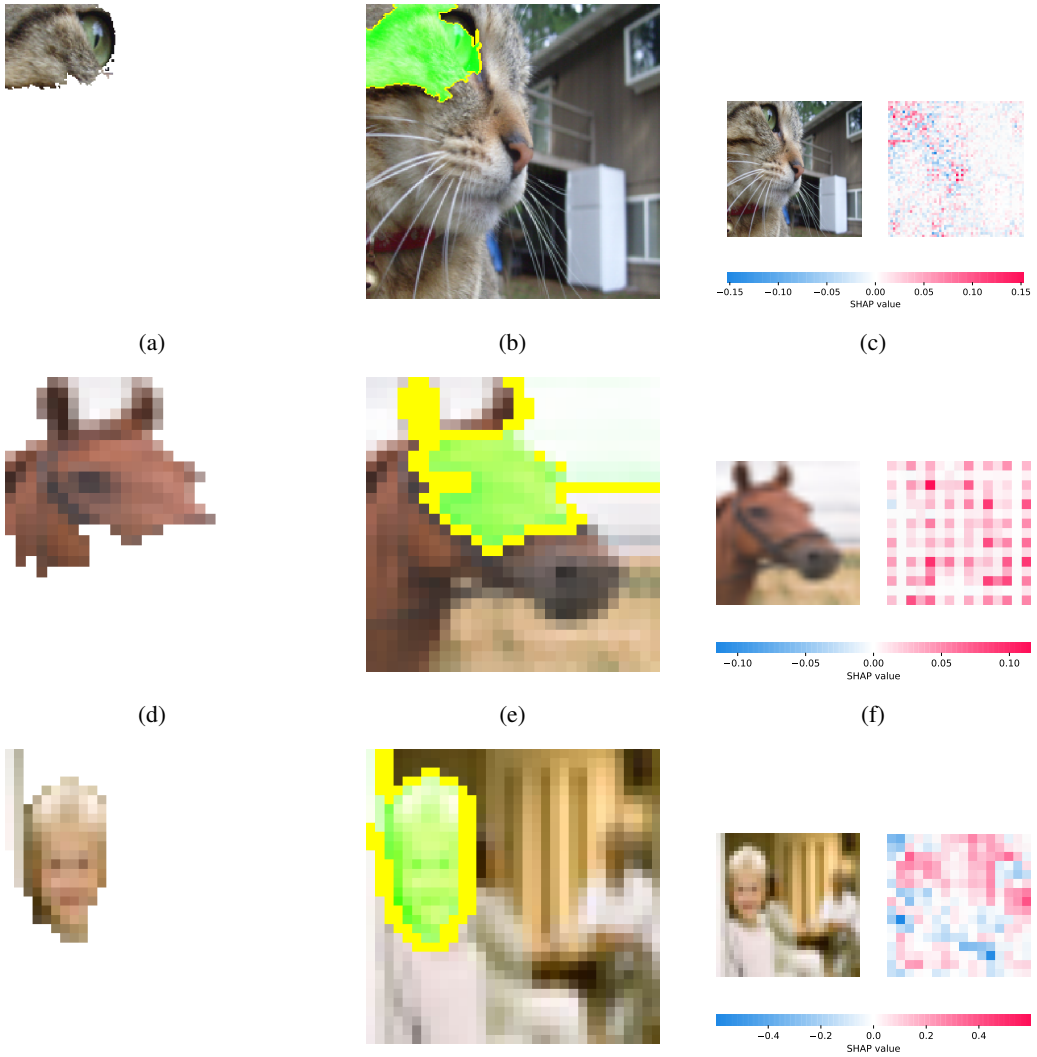


(a) Image of a deer (b) Non-robust AlexNet (c) Robust AlexNet

Figure 16: Comparison between explanations produced by AXAI ($K = 10$) for AlexNet trained on CIFAR10 with and without adversarial training

F Benchmark Tests

We test our algorithm against LIME and SHAP. We use “Gradient Explainer" in SHAP, which integrates the f Integrated gradients algorithm with SHAP. Fig. 17 shows some sample comparisons among the 3 algorithms for 3 cases: 1) AlexNet trained on CIFAR10, 2) ResNet34 trained on ImageNet, 3) VGG16 trained on CIFAR100. PGDM with 20 iterations is used in our algorithm. For ImageNet, explanations for a sample test picture belonging to “Egyptian cat" are shown in Fig. 17a, Fig. 17b, and Fig. 17c. One can see the similarity between the explanations. The explanations produced by the 3 algorithms focused on the upper left of the image which contains the eyes of the “Egyptian cat." Both LIME and our algorithms point to the same segment as explanations. SHAP (Gradient Explainer) locates pixels of interest. The important pixels shown in this case aligns with the results of LIME and AXAI. Since the default image segmentation parameters LIME chooses do not allow for a suitable number of segments for explanation for CIFAR10 and CIFAR100 due to the resolutions of images, we lowered the Kernel size parameter to 1. The default Kernel size parameters LIME uses for QuickShift is too large for low-resolution images. As we mentioned before, this leads to a few very large segments in the image and neglects all the granular details in the image. For CIFAR10, both our approach and LIME capture the upper portion of the head of the horse including the ears and eyes (Fig. 17d, Fig. 17e). The results of SHAP point out the important pixels located on the head, the nose and some pixels in the background (Fig. 17f). For CIFAR100, the explanations produced by the 3 algorithms are once again highly similar (Fig. 17g, Fig. 17h, and Fig. 17i). One can see that in many cases, pixel explanations do not serve as the best solution. Without the segments, it is hard to grasp the meaning behind explanations, this is because the human brain tends to comprehend image segments better than individual pixels.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 17: Comparisons between our adversarial explainability approach (Left Column), LIME (Middle Column), and SHAP (Right Column). LIME parameters: number of perturbed samples $N = 1000$, number of features $M = 5$. First row: ResNet34 trained on ImageNet, Second row: AlexNet trained on CIFAR10, Third row: VGG16 trained on CIFAR100