"Is your explanation stable?": A Robustness Evaluation Framework for Feature Attribution

Yuyou Gan*
Zhejiang University
ganyuyou@zju.edu.cn

Yuhao Mao*
Zhejiang University, ETH Zürich
yuhaomao@zju.edu.cn

Xuhong Zhang✉
Zhejiang University
zhangxuhong@zju.edu.cn

Shouling Ji ✉
Zhejiang University
sji@zju.edu.cn

Yuwen Pu
Zhejiang University
yw.pu@zju.edu.cn

Meng Han
Zhejiang University
mhan@zju.edu.cn

Jianwei Yin
Zhejiang University
zjuyjw@zju.edu.cn

Ting Wang
The Pennsylvania State University
inbox.ting@gmail.com

ABSTRACT

Neural networks have become increasingly popular. Nevertheless, understanding their decision process turns out to be complicated. One vital method to explain a models' behavior is feature attribution, *i.e.*, attributing its decision to pivotal features. Although many algorithms are proposed, most of them aim to improve the faithfulness (fidelity) to the model. However, the real environment contains many random noises, which may cause the feature attribution maps to be greatly perturbed for similar images. More seriously, recent works show that explanation algorithms are vulnerable to adversarial attacks, generating the same explanation for a maliciously perturbed input. All of these make the explanation hard to trust in real scenarios, especially in security-critical applications.

To bridge this gap, we propose *Median Test for Feature Attribution* (MeTFA) to quantify the uncertainty and increase the stability of explanation algorithms with theoretical guarantees. MeTFA is method-agnostic, *i.e.*, it can be applied to any feature attribution method. MeTFA has the following two functions: (1) examine whether one feature is significantly important or unimportant and generate a MeTFA-significant map to visualize the results; (2) compute the confidence interval of a feature attribution score and generate a MeTFA-smoothed map to increase the stability of the explanation. Extensive experiments show that MeTFA improves the visual quality of explanations and significantly reduces the instability while maintaining the faithfulness of the original method. To quantitatively evaluate MeTFA's faithfulness and stability, we further propose several robust faithfulness metrics, which can evaluate the faithfulness of an explanation under different noise settings. Experiment results show that the MeTFA-smoothed explanation can significantly increase the robust faithfulness. In addition, we use two typical applications to show MeTFA's potential in the applications. First, when being applied to the SOTA explanation method to locate context bias for semantic segmentation models, MeTFA-significant explanations use far smaller regions to maintain 99%+ faithfulness. Second, when testing with different explanation-oriented attacks, MeTFA can help defend vanilla, as well as adaptive, adversarial attacks against explanations.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**.

KEYWORDS

Explanation of AI; Robustness of Explanation; Hypothesis Testing

ACM Reference Format:
Yuyou Gan, Yuhao Mao, Xuhong Zhang, Shouling Ji, Yuwen Pu, Meng Han, Jianwei Yin, and Ting Wang. 2022. "Is your explanation stable?": A Robustness Evaluation Framework for Feature Attribution. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22), November 7–11, 2022, Los Angeles, CA, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3548606.3559392

1 INTRODUCTION

The extraordinary performance of deep neural networks (DNNs) has led us to the era of deep learning [3][19][40][53]. While these networks achieve human-level performance on various tasks, there are many security crises in DNNs [24][28][15][27], which prevent users from fully trusting the models' output, raising concerns for sensitive applications like autonomous driving. Many works are proposed to make DNNs more safe and reliable [51][12][25], one of which is to explain the action of model. By interpreting the black-box model, we can detect biases and anomalies, and gain insights to improve the model.

Feature attribution, which explains the model at instance level, is a popular method to explain neural networks [31][41][17][23]. These explanations give each feature a feature score representing its contribution to the model's output. Features with high contribution scores support the decision of the model, and thus we call them

*Yuyou Gan and Yuhao Mao contributed equally. Xuhong Zhang and Shouling Ji are the corresponding authors.

CCS '22, November 7–11, 2022, Los Angeles, CA, USA

ACM ISBN 978-1-4503-9450-5/22/11…$15.00
https://doi.org/10.1145/3548606.3559392

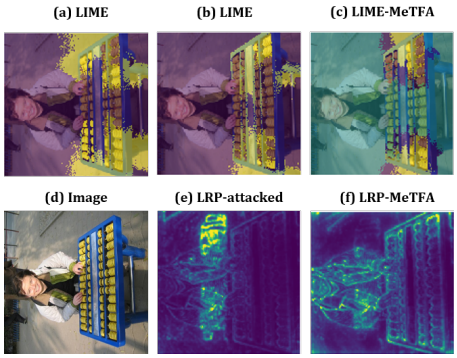

Figure 1: Example explanations on VGG16. (a) and (b) are the explanations by two independent runs of LIME [34], a black-box explanation method. (c) is the MeTFA-significant LIME explanation, where the yellow area is significantly important, the green area is significantly unimportant, and the rest region is insignificant. (d) is the seed image of manipulation attack [11], an attack that keeps the model's prediction unchanged but manipulates the explanation of the prediction. (e)Attack[11] for LRP[4] shows the explanation of the LBP for the adversarial image generated by manipulation attack from (d), where the explanation is totally irrelevant to the prediction abacus. (f) is the MeTFA-smoothed explanation of the same adversarial image, where a plausible explanation is recovered.

supportive features. For instance, in the image domain, attribution map is a common form of feature attribution. From the attribution map, users can see which parts of the image are relied on by the model to make the prediction. In addition, feature attribution methods help to create better models. For example, recent works use feature attribution to debug errors [18], detect adversarial examples [50], and check context bias [20].

However, the feature attribution methods used to explain neural networks are still facing a reliability crisis. The feature attribution methods can be roughly classified into two categories: black-box methods and white-box methods. Black-box methods only have access to the input and output of a network. Generally, such methods apply various forms of sampling, which leads to uncertainty in the explanation [18][30][34]. For example, as shown in Figure 1 (a) and (b), LIME [34] gives different attribution maps for the same image in independent runs. On the contrary, white-box methods mainly use deterministic information, such as gradient [4][38] and activation maps [52][42], to generate attribution maps, thus guarantee a deterministic explanation for the same input. However, they still have problems due to instability. For example, gradient-based methods produce drastically different attribution maps for similar inputs [39]; optimization-based methods sometimes generate nonsensical or unexpected attribution maps due to the instability of the non-robust features [14]. Moreover, explanations with high sensitivity may be more vulnerable to adversarial attacks [46]. For example, in Figure 1 (d) and (e), by adding human-imperceptible noise, the attacker can manipulate an attribution map arbitrarily. These phenomena greatly reduce our trust in explanation algorithms.

As illustrated above, existing feature attribution methods give uncertain results due to the sampling process or the non-robust features. Therefore, to make the explanations reliable, a method to reduce and quantify the uncertainty involved in the explanation is required. To increase the stability of the explanations, a promising way is to sample inputs from the neighborhood of the original input and average all these explanations. When the explanation is Gradient [38], this method is known as SmoothGrad [39]. Therefore, by the law of large numbers, the smoothed explanation essentially converges to the expectation of the distribution of the neighborhood explanations of the original input, thus mitigating the uncertainty to some degree.

Although this method, along with its modification [47], are designed only to produce stable results, it can be further extended to quantify the uncertainty of the explanation by computing the standard deviation of the samples. Specifically, by the central limit theorem, they can be extended to use the mean and standard deviation to derive an asymptotic confidence bound of the explanation. Details on how to derive the bound are included in Section 7. However, the correctness of these bounds requires the law of large numbers, *i.e.*, they are only valid when the number of samples is extremely large. This brings heavy computational overheads because sampling explanations is computationally expensive. In addition, when the number of samples is small, the mean value is sensitive to abnormal extreme values. Since the underlying distribution of explanations is unknown, it is difficult to get any theoretical guarantees for the confidence bounds with such methods when the number of samples is small. Therefore, how to efficiently quantify the uncertainty of the explanations remains unresolved.

Our designs. To overcome the above challenges, we propose *Median Test for Feature Attribution* (MeTFA). Instead of generating a single score for each feature, MeTFA takes a novel perspective: a reliable explanation should include the attribution score map together with the confidence interval and the significance of the scores. The core idea of MeTFA is to sample around the original data and conduct the hypothesis test on the samples' explanations to establish theoretical guarantees. To tackle the problem of unknown distribution and the impact of the abnormal extreme values, MeTFA approximates the median of the explanation distribution instead of the expectation. In this way, MeTFA converts the unknown distribution to a Bernoulli distribution on a specific statistics without approximating the normal distribution by the central limit theorem, thus allowing it to get exact confidence bounds with a small number of samples. MeTFA considers the following two scenarios: (1) the users only want to know which features are significantly important or unimportant with regard to a threshold of the feature score, and (2) the users wish to know the confidence bounds for the feature score. The first scenario is more general because many explanation algorithms, such as LIME [34], only provide a discrete binary score as the feature attribution, while the second scenario requires a continuous feature score in the range of $[0, 1]$. For the first scenario, given a user-interested feature score threshold and a confidence level α, we design the one-sided MeTFA to generate the MeTFA-significant map. This map shows whether one feature is significantly important (the median of the explanation distribution is higher than the threshold) or unimportant (the median of the explanation distribution is lower than the threshold) with regard to the confidence level α. For the second scenario, we design the two-sided MeTFA to compute the median's α-confidence interval for the features. Further, by averaging the sampled feature scores in the confidence interval,

MeTFA generates the MeTFA-smoothed explanation as the approximation of the median. This is different from SmoothGrad which averages over all the sampled scores. In Figure 1 (c) and (f), the results of LIME [34] with one-sided MeTFA and LRP [4] with two-sided MeTFA show that MeTFA can reveal the significantly important features and defend against the explanation-oriented attacks. In addition, we prove by theoretical and empirical findings that the variance of the MeTFA-smoothed explanation shrinks to 0, which suggests the correctness of MeTFA. Moreover, the variance shrinks with a same or faster speed than SmoothGrad, which suggests that MeTFA-smoothed explanations are more stable and efficient.

Evaluations. In the image domain, we evaluate our method on representatives of the four main types of explanation methods: (1) gradient-based method: Gradient [38], (2) sample-based method: RISE [30] and LIME [34], (3) optimization-based method: IGOS [33], and (4) activation-based method: ScoreCAM [42]. We use the popular metrics *insertion*, *deletion* [30] and *overall* [48] as the faithfulness metrics and *standard deviation* (std) of feature scores as the stability metric. Further, we propose the *robust insertion*, *robust deletion* and *robust overall* metrics to measure the ability of explanation to locate robust features. The results show that MeTFA slightly affects the faithfulness for the orignal input but significantly increases the robust faithfulness. In addition, we show that MeTFA is better than SmoothGrad in stability. In the NLP domain, we evaluate MeTFA with LEMNA [18], the SOTA explanation alogorithm targeting RNN. We use *feature deduction test*, *feature augmentation test* and *synthetic test* proposed in [18] as the faithfulness metrics and take *std* and the *overlap of top n features* as the stability evaluation metrics. Similar to the image domain, we propose three corresponding robust faithfulness metrics. Experiment results show that MeTFA can improve all the faithfulness, stability and robust faithfulness metrics for LEMNA.

Applications. To illustrate the potential of MeTFA in practice, we apply one-sided MeTFA and two-sided MeTFA, respectively, to two applications closely related to security: detecting the context bias in the semantic segmentation and defending the explanation-oriented adversarial attack. We apply the one-sided MeTFA to GridSaliency [20], the SOTA method to locate context bias for semantic segmentation models. The results show that, in the environment with common noise distributions, MeTFA can greatly reduce the context bias region while maintaining 99%+ faithfulness, which suggests the potential of MeTFA to detect context bias in the real world. Moreover, we demonstrate that two-sided MeTFA can defend both the attacks that produce wanted explanations while keeping model's predictions unchanged and the attacks that produce unchanged explanations for wanted predictions.

Contributions. (1) We propose a novel perspective: a reliable explanation should include not only the attribution score map but also the confidence interval and the significance of the scores. (2) We propose a framework MeTFA to quantify the uncertainty and increase the stability of the feature attribution algorithm with theoretical guarantees. (3) We propose a series of robust metrics which consider the neighborhood of the input instead of a single point. Experimental results show that MeTFA increases the stability while maintaining the faithfulness. (4) The application of MeTFA on detecting the context bias in semantic segmentation and defending against the adversarial examples with the explanation-oriented attack shows its great potential in real practice.

2 RELATED WORK

Visual explanation. Visual explanation methods can be divided into white-box methods and black-box methods. White-box explanations are free to use all the information about the model, such as architecture and parameter. They can be roughly categorized into three groups: gradient-based, activation-based and optimization-based. Gradient-based methods [5] [38] use gradient information to generate the feature importance of pixels, known as saliency maps. These methods are typically fast but may render volatile explanations due to the sensitivity of the gradients of input samples [39]. Activation-based methods [52] [37] [42] address this problem by using the activations of convolutional layers instead. They apply upscaled linear combination of some layers' output as the explanation and find that it usually highlights the region of the correct object and is more stable. Optimization-based methods [49] [41] [14] [33] do not generate feature importance for all the pixels but highlight a small area of interest using optimization. They typically generate high-quality explanations but are much slower, as the optimization process requires multiple forward and backward propagations. Black-box explanations [30] [34] [32] only require the access to the input and output of the model. They randomly sample some features, modify these features (*e.g.*, LIME sets them to 0), put the modified inputs into the model and explain using the outputs (*e.g.*, LIME uses a linear model to fit the outputs and the modified inputs). It can be found that MeTFA and sample-based explanation methods use sampling in different ways and for different purposes. MeTFA uses the common noise in real life to sample inputs around the original input and then conduct hypothetical testing on their explanations to increase the stability of explanation methods in the real world.

Evaluation of visual explanation. Šikonja *et al*. [35] summarized the required properties of explanations. Some of them are: (1) faithfulness to the model, (2) stability of the explanation when the input is slightly perturbed, and (3) comprehensibility of the explanation to humans. Unfortunately, current explanation algorithms are not satisfactory at these properties. Ghorbani *et al*. [16] found that explanations could be largely affected by adversarial perturbations which do not change the model's prediction. In addition, Kindermans *et al*. [22] found that for two networks with provably same explanations, the explanations for the two networks produced by current explanation algorithms are different. Moreover, Adebayo *et al*. [1] found that when the network is gradually randomized, many algorithms do not produce randomized explanations. Instead, they highlight "edge pixels", which is similar to edge detectors. By evaluating recent explanation algorithms, these works show that there is a large gap to fulfill in the explanation algorithms.

Post hoc improvement on the explanation. There are some attempts aiming at improving the stability of explanations by post hoc improvements. They are built on a stability assumption that the explanations should not vary greatly for similar inputs. Smilkov *et al*. [39] first showed that the gradient-based explanations were vulnerable to small perturbation to the input. They found by experiment that adding Gaussian noises to the input and averaging their explanations would provide an explanation with better visual quality. This method is called *SmoothGrad*. To understand why SmoothGrad

works, Yeh *et al*. [47] proposed a theoretical framework which justifies that an extension of SmoothGrad using kernel functions could improve the faithfulness of explanations as well. Agarwal *et al*. [2] prove that SmoothGrad and a variant of LIME converge to the same explanation in expectation. Our work targets post hoc improvement of stability. While previous works only proposed usable heuristics, our paper takes it further and first makes statistical tests possible for feature attribution.

3 MEDIAN TEST FOR FEATURE ATTRIBUTION

In this section, we first define some related concepts. Then we introduce the one-sided MeTFA and two-sided MeTFA, respectively. All proofs are included in Appendix A.1 due to the space limitation.

3.1 Overview

MeTFA can be applied to any algorithm that explains the prediction by evaluating feature importance. Let the prediction function be $F : (X_i)_{i\in S} \to O$, where S is the input feature set, X_i is an individual feature and O is the model's output. Then an explanation algorithm which assigns features with importance can be denoted as $E : (F, (X_i)_{i\in S}) \to [0,1]^{|S|}$, where $|S|$ is the number of features included in S. For example, in image classification, S is the set of all pixels, and $|S|$ equals to $w \times h$, where w is the width, and h is the height of the image.

Following Smilkov *et al*. [39], MeTFA is built upon the axiom that the generated explanation should be similar if the input and the output of the prediction function are similar. Formally, let $\mathbb{P}$ be the noise set under which we presume the prediction function is robust, *i.e*., O changes little for inputs after adding some noises sampled from $\mathbb{P}$. Similar to Smilkov *et al*. [39] and Yeh *et al*. [47], the first step of MeTFA is to sample noises from $\mathbb{P}$, add them to the original input, pass these noisy inputs all through the explanation algorithm and get explanations $\{e_i\},\ i = 1, \dots N.$, which subject to some unknown distribution $\mathbb{D}$. We call $\{e_i\}$ the sampled explanations. e_{ij} is the sampled feature score in e_i for feature j.

We tackle the unknown distribution by converting $\mathbb{D}$ on the explanations to a Bernoulli distribution on a specific statistics called counting variable, based on the following two key properties of the median V:

- If most of the samples are greater than a fixed value h, then V is more likely to be greater than h, vice versa.
- For any continuous distribution $\mathbb{D}$ and $x \sim \mathbb{D}$, we have $P(x \geq V) = 0.5$ and $P(x \leq V) = 0.5$. Therefore, for any $h \geq V$, $P(x \geq h) \leq 0.5$, vice versa.

Based on the first property, we define the *counting variable* $ct_j(h) = \sum_{i=1}^{N} I(e_{ij} \geq h)$, where I is the indicator function. Note that $I(e_{ij} \geq h)$ follows a Bernoulli distribution with parameter $q_j(h) = P(e_{ij} \geq h)$. In addition, e_{ij}, $i = 1, 2, ..., N$, is independent for any fixed j. Therefore, $ct_j(h) \sim B(N, q_j(h))$, where B is a Binomial distribution. Then we use ct_j as the test statistic to design hypothesis testing.

The second property provides a way to estimate the p-value for continuous explanation. In practice, $\mathbb{D}$ may be discrete for some explanation methods, e.g., LIME. However, we can easily approximate any discrete distribution by a continuous distribution to any precision. Specifically, we add a very small continuous noise, *e.g*., $\mathcal{N}(0, 10^{-6})$, to $\mathbb{D}$. For example, consider LIME which gives 0-1 map of the pixels as the explanation. The distribution of LIME's explanation on noisy inputs is discrete, as it can only be 0 or 1. Further, assume that given a particular input and a particular pixel, the probability of being 0 is 0.6, and thus the median of the distribution of LIME's explanation is 0. Then, the probability of sampling a value greater than 0 is only 0.4, not the 0.5 that we utilize. In this case, MeTFA has an assumption violation. However, if we add a small noise to LIME's explanation, say $\mathcal{N}(0, 10^{-6})$, then the LIME's explanation becomes a sharp bi-modal distribution which is concentrated around 0 and 1. The modified distribution is continuous, and thus the assumption that the probability of a sample greater than the median equals to 0.5 holds, which makes MeTFA applicable. In addition, as long as the modification noise is continuous and very small, this approximation only possibly changes the median by a tiny difference, thus the result is not affected. In particular, applying $\mathcal{N}(0, 10^{-6})$ or $\mathcal{N}(0, 10^{-8})$ does not make a difference for LIME. Note that we add small perturbation to the sampled explanations here rather than to the input, and this only aims at making the distribution of $\{e_i\}$ continuous. Therefore, for simplicity, we assume $\mathbb{D}$ is continuous in the paper.

In the following, we describe two different processing of $\{e_i\}$ to perform the importance test (whether V is greater or smaller than h) in Section 3.2 and bound test (the interval that V most likely falls into) in Section 3.3, respectively.

3.2 One-sided MeTFA

One-sided MeTFA is to test $H_0 : V < h$ or $H_0 : V > h$ for some fixed h. This can be broken down into two steps. First, convert $\mathbb{D}$ to a Bernoulli distribution. Second, conduct the test based on the Bernoulli distribution. Thus, in the following, we first derive how to construct the Bernoulli distribution and then conduct the statistics test.

We have derived in Section 3.1 that the counting variable $ct_j(h) \sim B(N, q_j(h))$, where $q_j(h) = P(e_{ij} \geq h)$. Using this property, we are able to derive Proposition 1.

Proposition 1. *Suppose V_j is the median of $\mathbb{D}$ for feature j. Then, $P(ct_j(h) \geq m) \leq \sum_{i=m}^{N} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$ for $V_j \leq h$, where $p_* = \min(0.5, i/N)$, and N is the number of sampled explanations. Similarly, $P(ct_j(h) \leq m) \leq \sum_{i=0}^{m} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$ for $V_j \geq h$, where $p_* = \max(0.5, i/N)$. The proof is in Appendix A.1.1.*

Using Proposition 1, we can get Theorem 1 which enables us to perform the one-sided MeTFA.

Theorem 1. *Suppose we observe $ct_j(h) = k$. Then, the p-value of $H_0 : V_j \leq h$ is $\sum_{i=k}^{N} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$, where $p_* = \min(0.5, i/N)$. Similarly, the p-value of $H_0 : V_j \geq h$ is $\sum_{i=0}^{k} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$, where $p_* = \max(0.5, i/N)$. The proof is in Appendix A.1.2.*

Therefore, to test whether V_j is greater or smaller than h, we first count how many times e_j is greater than h, compute the p-values according to Theorem 1, and then compare the p-values with user-interested confidence level α. From the procedure described above, we can see that the total complexity is $O(N)$ for the one-sided MeTFA.

3.3 Two-sided MeTFA

Two-sided MeTFA is to test $H_0 : V = h$ for some fixed h. Under H_0, we have $q_j(h) = 0.5$ and a too large or too small $ct_j(h)$ is rare. Using this property, we are able to derive Proposition 2.

Proposition 2. *$P(ct_j(h) \le m_1) + P(ct_j(h) \ge m_2) \le \sum_{i\in\{0,\dots,m_1\}\cup\{m_2,\dots,N\}} 0.5^N \times \binom{N}{i}$ for $V_j = h$. The proof is in Appendix A.1.3.*

Proposition 2 directly suggests the two-sided MeTFA, as shown in Theorem 2.

Theorem 2. *Suppose we observe $ct_j(h) = k$. Let $k_1 = \min(k, N-k)$ and $k_2 = \max(k, N-k)$. Then, the p-value of $H_0 : V = h$ is $\sum_{i\in\{0,\dots,k_1\}\cup\{k_2,\dots,N\}} 0.5^N \times \binom{N}{i}$. The proof is in Appendix A.1.4.*

A direct application of Proposition 2 gives the confidence interval of V as well. It is shown in Theorem 3.

Theorem 3. *Let $k_1 = \text{argmax}_k\{\sum_{i=0}^{k} 0.5^N \times \binom{N}{i} \le \frac{\alpha}{2}\}$ and $k_2 = N - k_1$. Let h_{1j} be the k_1-th smallest in $\{e_{ij} \mid i = 1, \dots, N\}$ and h_{2j} be the k_2-th smallest. Then (h_{1j}, h_{2j}) is a $1-\alpha$ confidence interval for V_j. The proof is in Appendix A.1.5.*

Therefore, to test whether V_j is equal to h, we first count how many times e_j are greater than h, compute the p-values according to Theorem 2, and then compare the p-values with custom confidence levels. To obtain the confidence intervals, we need to find the maximum k_1 that makes $\sum_{i=0}^{k_1} 0.5^N \times \binom{N}{i}$ smaller than $\alpha/2$, compute k_2 by $N - k_1$, and then get the interval from the sorted $\{e_{ij}\}$. We name the map consisting of h_{1j} the *lower bound map* and the map consisting of h_{2j} the *upper bound map*. From the procedure described above, we can see that the complexity is $O(N)$ for the two-sided MeTFA and $O(N \log N)$ for computing the confidence interval. The pure test takes a very short time compared to the sampling process, which is bottlenecked by the speed of the explanation algorithm. However, the sampling process can be fully parallelized to take a constant time.

4 METFA-BASED ATTRIBUTION MAP

In this section, based on MeTFA, we first design two kinds of maps, named *MeTFA-significant map* and *MeTFA-smoothed map*, to point out the significant important (unimportant) supportive features and quantify the stability of explanation, respectively. Then, we give the lower bound of the number of samples to achieve a user-interested confidence level α.

4.1 MeTFA-Significance Map

While the attribution maps that show every feature's importance are informative, in many cases, we only want to know what features are important and what features are unimportant. For example, when we explain the prediction of an image classification model to laypersons, they only want a subregion of the input image highlighting the important features, which motivates us to develop *MeTFA-Significance Maps* to highlight the important features and unimportant features. We use the task of image classification to show the core idea.

Formally, when trying to figure out the "important" and "unimportant" features, we are actually classifying these features into two groups. Therefore, we first do a global one-dimensional two-group clustering using *Jenks natural breaks algorithm* [21] for the attribution values in the sampled $\{e_{ij}\}$. This allows us to find the optimal h for the two groups, *i.e.*, h is recommended to be the break threshold of these two groups. Since the "best" h varies from image to image, compared with previous works, which set a fixed threshold h [10] based on their requirements, our clustering approach can select a better h adaptive to the image. Next, we perform one-sided MeTFA for every feature with regard to the threshold h. The features whose scores are significantly greater than h are classified as important, and significantly smaller than h are classified as unimportant. Others are in the between, denoted as "undecided", meaning that they are neither significantly greater than h nor significantly smaller than h. In the case of image classification, we paint the important features in yellow, the unimportant features in dark green, and the undecided features in dark purple. An example is presented in Figure 2.

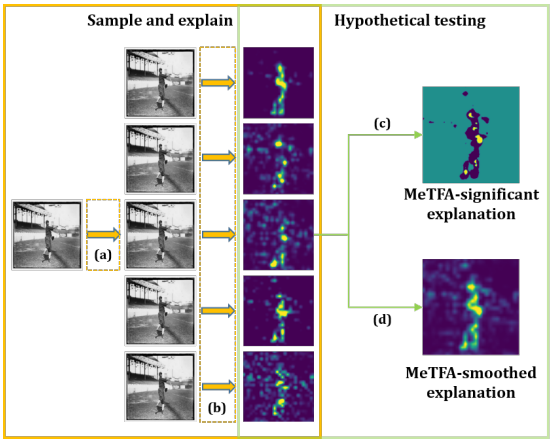

Figure 2: The overview of MeTFA. First, in (a), we add some noise from a specific distribution $\mathbb{P}$ (*e.g.*, Normal, Uniform) to the clean image. Then, in (b), we generate an explanation for each noisy image using an existing attribution method (*e.g.*, RISE, IGOS). Finally, we use the one-sided MeTFA (in (c)) to generate the MeTFA-significant map and use the two-sided MeTFA (in (d)) to generate the MeTFA-smoothed map.

4.2 MeTFA-smoothed Map

Two-sided MeTFA can deduct a stabilization method to stabilize explanation algorithms. In the following, we first introduce the stabilized explanation named MeTFA-smoothed map, and then, we theoretically prove that the variance of theMeTFA-smoothed explanation shrinks to 0 with a same or faster speed than the SmoothGrad explanation.

The MeTFA-smoothed explanation uses the mean of the explanations only included in the confidence interval rather than all of the sampled explanations. The formal definition is shown in Definition 1.

Definition 1. *Suppose we have the sampled explanations $\{e_i\}$, and we have computed k_1 and k_2, respectively. Let $e_{(a)j}$ be the a-th smallest element in $\{e_{ij}\}$. Then the MeTFA-smoothed explanation is calculated as follows: the attribution score of feature j is defined to be $\sum_{a=k_1+1}^{k_2-1} e_{(a)j}/(k_2 - k_1 - 1)$. We denote it as $\mathcal{S}_j$.*

The MeTFA-smoothed explanation has two important properties. They are summarized in Theorem 4.

Theorem 4. *$\mathcal{S}_j$ possesses the following properties. The proof is in Appendix A.1.6:*

(1) *$\mathcal{S}_j$ converges to V_j when N is sufficiently large.*
(2) *Under the mild assumption that $f_e(V_j) > 0$, where f_e is the PDF of e,* $\mathrm{Var}(S_j)$ *converges to zero with a speed of at least* $O(1/N)$.

Theorem 4 tells us that the MeTFA-smoothed explanation is a consistent estimator for V_j, the median of the distribution of the sampled explanations. In addition, the variance of the MeTFA-smoothed explanation shrinks to 0, which suggests the correctness of MeTFA. Moreover, the variance shrinks with a same or faster speed than SmoothGrad, which suggests that MeTFA-smoothed explanations are more stable and efficient. In addition, to visualize the uncertainty in the explanations, we define the upper bound map and lower bound map to be the map visualizing the corresponding upper bound and lower bound. By comparing these two bounds, one can easily find the most uncertain features and locate the almost certain features.

We provide Algorithm 1 in the Appendix to demonstrate how to compute the MeTFA-significant explanation, MeTFA-smoothed explanation, upper bound map and lower bound map in more details.

4.3 Number of Sampled Explanations

The number of sampled explanations N is a critical hyperparameter for MeTFA. In fact, N is closely related to the confidence level α of the demand. For one-sided MeTFA, as shown in Theorem 1, for a fixed N, $p \geq \binom{N}{0} \times 0.5^0 \times 0.5^N = 0.5^N$. In order to reject H_0, $\alpha \geq p \geq 0.5^N$. Thus, the lower bound of N to achieve a given α is $\lceil -\log_2 \alpha \rceil$. Similarly, for two-sided MeTFA, as shown in Theorem 2, for fixed N, $p \geq \binom{N}{0} \times 0.5^N \times 2 = 0.5^{N-1}$. In order to reject H_0, $\alpha \geq p \geq 0.5^{N-1}$. Thus, the lower bound of N to achieve a given α is $\lceil -\log_2 \alpha \rceil + 1$. However, these are the minimum number of samples required, and we recommend to use more samples whenever the computational cost of the sampling explanations is acceptable.

5 EXPERIMENTS

In this section, we first explain the experiment settings in details. Then, we demonstrate the quality of MeTFA explanations from three perspectives: visualization, stability and faithfulness. Finally, we discuss the impact of important hyperparameters for MeTFA.

5.1 Settings

We evaluate MeTFA on the image classification and the text classification task because most feature attribution methods target these two tasks. The evaluation settings is as follows:

5.1.1 Datasets and Models. For the image classification task, we use ILSVRC2012 validation set [36] as the source dataset, because it is the most evaluated dataset among explanation methods. We use the pre-trained VGG16, Resnet50 and Densenet169 from PyTorch [29] as the models to be explained.

For the text classification task, we choose the dataset from Toxic Comment Classification Challenge[1] which contains 159,571 training texts and 63,978 testing texts with six toxic comment classes including toxic, severe toxic, obscene, threat, insult and identity hate. We train a bidirectional LSTM concatenated with two fully connected layers on the training set, which achieves an accuracy rate of 97.46%. Since a sentence consisting of too many words will greatly increase the number of samples required by LENMA, in the following experiment, we only use the sentences of length between 40 and 80. We extensively evaluate MeTFA on the IMDb Movie Reviews dataset [26]. We split the 50000 reviews into training set (containing 40000 reviews) and test set (containing 10000 reviews), and the same model is applied. The trained model achieves an accuracy rate of 88.39% on the test set.

5.1.2 Explanation Algorithms. In the image domain, we apply MeTFA to four types of mainstream feature attribution methods, including Gradient (the most classic gradient-based method), RISE and LIME (two most popular sample-based methods), IGOS (the SOTA optimization-based method) and ScoreCAM (the SOTA CAM-based method). In the text domain, we apply MeTFA to LENMA, the SOTA explanation method designed for RNN.

5.1.3 Metrics. We introduce the metrics to evaluate the stability of explanations and the faithfulness of explanations

Stability. The standard deviation is a good choice to measure the variety of the output. To measure the stability of a feature attribution method under noises, we use the *mstd* (short for the mean of *std*) metric, defined to be the mean of the standard deviation of explanations on noisy inputs sampled from the neighborhood of the original input. The formal definition of mtsd is as follows:

$$\text{mstd} = Mean_{I \in D, i \in S}(std_{n \sim \mathbb{O}_n}(E(I+n)_i))$$

where $E(\cdot)_i$ returns the attribution score of feature X_i and D is the test data set. The default number of noisy inputs is set to 10, *i.e.*, for every image, we sample 10 noises n from $\mathbb{O}_n$. By the definition, a lower mstd value means a more stable explanation.

Faithfulness. The faithfulness metric is used to measure whether the features highlighted by an attribution map support a model's prediction. An explanation is called faithful if the generated attribution maps highlight the supportive features. In our experiments, we use two kinds of metrics to evaluate the faithfulness. One is the most popular metric used in the image and text domain proposed in the previous work, and the other one is our proposed more robust faithfulness metric.

In the image domain, *insertion*, *deletion* [30] and *overall* [48] are commonly used to estimate the faithfulness of an attribution map. These metrics aim to measure whether the features highlighted by an attribution map support a model's prediction. If an attribution map is faithful to the model, then removing the pixels with the highest values from a full image will cause a big decrease on the predicted score and conversely, inserting the pixels with the highest values into a blank image will cause a big increase on the predicted score. To formally illustrate this property, let I be the original image, $f_c(\cdot)$ returns the predicted score of label c and M is the attribution map given by a explanation algorithm for I. Then, the (normalized) insertion and deletion are defined as follows:

$$insertion(I, M) = \frac{1}{f_c(I)} \int_0^{100} f_c(I_n^{M+}) dn$$

[1] https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge/data

$$deletion(I, M) = \frac{1}{f_c(I)} \int_0^{100} f_c(I_n^{M-}) dn$$

where I_n^{M+} keeps the $n\%$ pixels in I with top attribution scores, and I_n^{M-} deletes the $n\%$ pixels in I with top attribution scores. A more faithful attribution map can highlight the supportive features more accurately and thus keeping the same number of pixels can get a higher predicted score, resulting in a higher insertion value. Similarly, a more faithful attribution map gets a lower deletion value. To specifically show the process, we add an example in the appendix (Figure 10). Besides, sometimes insertion and deletion give the contradictory results. In such situation, overall, which is equal to insertion minus deletion, is used to evaluate the faithfulness. However, the faithfulness metrics considering only a single point suffers from the effect of non-robust features. Therefore, we take the neighborhood of the input into consideration. Specifically, we propose *robust insertion* (RI), *robust deletion* (RD) and *robust overall* (RO) to further evaluate the faithfulness for the neighborhood of the clean image, which are as follows.

$$RI(M, I, \mathbb{O}_n) = \mathbb{E}_{n\sim\mathbb{O}_n}(insertion(I + n, M))$$

$$RI(M, I, \mathbb{O}_n) = \mathbb{E}_{n\sim\mathbb{O}_n}(insertion(I + n, M))$$

where $\mathbb{O}_n$ is the distribution of noise. The intuition behind the RI (RD) metric is that if the attribution map finds the robust supportive features, then after adding a small random noise, the features will still keep supporting and thus having a high insertion score or low deletion score on the noisy images. Therefore, a higher robust faithfulness means the explanation can locate the robust features more precisely. In practice, we use 10 samples to approximate the expectation. Correspondingly, RO is defined as RI minus RD. By the definition, a lower value of deletion or RD suggests a more faithful explanation while a higher value of insertion, overall, RI or RO suggests a more faithful explanation.

In the text domain, we use *Feature Deduction Test* (FDT), *Feature Augmentation Test* (FAT) and *Synthetic Test* (ST) proposed in LEMNA to estimate the faithfulness of an explanation. Similar to the image task, let T be the original text, $f_c(\cdot)$ returns the predicted score of label c and M is the attribution map given by a explanation algorithm for T. Then FDT, FAT, ST can be defined as follows:

$$FDT(T, M) = f_c(T_n^{M-})$$

$$FAT(T, M, T') = f_c(T' \circ T_n^{M+})$$

$$ST(T, M) = f_c(T_n^{M+})$$

where T_n^{M-} deletes the n words in T with top attribution scores, $T' \circ T_n^{M+}$ retains the n words with top attribution scores in T but replaces the other words by a randomly selected instance T', and T_n^{M+} only keeps the n words in T with top attribution scores. Similar to the image domain, we extend these three metrics to the robust faithfulness metrics, *i.e.*, RFDT, RFAT and RST, and use 10 samples for calculation. By the definition, a lower value of FDT or RFDT suggests a more faithful explanation while a higher value of FAT, ST, RFAT or RST suggests a more faithful explanation.

$$RFDT(T, M, \mathbb{O}_n) = \mathbb{E}_{n\sim\mathbb{O}_n}(FDT(T + n, M))$$

$$RFAT(T, M, \mathbb{O}_n, T') = \mathbb{E}_{n\sim\mathbb{O}_n}(FAT(T + n, M, T'))$$

$$RST(T, M, \mathbb{O}_n) = \mathbb{E}_{n\sim\mathbb{O}_n}(ST(T + n, M))$$

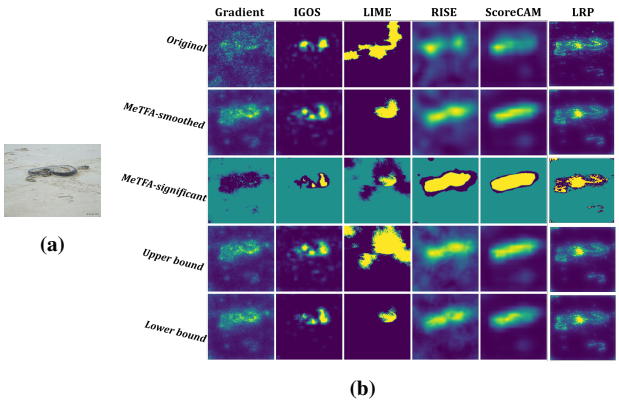

Figure 3: Visual examples for the original, the MeTFA-smoothed and the MeTFA-significant explanations. (a) is the input image. The columns of (b) show the explanations of RISE, IGOS, LIME, Gradient and ScoreCAM, respectively.

The roles of the noise distribution $\mathbb{P}$ and $\mathbb{O}_n$ are different. $\mathbb{P}$ is used to sample around the original data, which is a core step of MeTFA, while $\mathbb{O}_n$ is used to compute the robust metrics. We take RI as an example to show the difference more specifically and we provide its algorithm in Algorithm 2 in the appendix.

5.1.4 Default Settings. Unless otherwise specified, all the hyper-parameters are set as follows.

- The confidence level α. We set a common choice $\alpha = 0.05$.
- The number of sampled explanations N. As discussed in Section 4.3, the minimum N to achieve $\alpha = 0.05$ for MeTFA-significant map is 5. However, this leads to too few features being tested significant. Therefore, we choose $N = 10$ as the default.
- The number of samples for the sample-based methods. Typically, the more samples used by the sample-based method, the more stable the generated explanation would be. For LIME, we choose 1000 because this is the default in the LIME's open-source code. For RISE, we choose 1000 as well so that it is consistent to LIME since they are compared to each other in the image domain. For LEMNA, we use 500 and 2000 for the Toxic Comments dataset and 2000 for the IMDb Reviews dataset.
- For the other parameters, we use the same as the corresponding open-source code.

5.2 Quality of Visualization

In this part, we illustrate the visualization effect of MeTFA in the image domain. We use the pre-trained VGG16 network from the Pytorch and an image from the source dataset as an example. The predicted label of the image is "sea snake", which is consistent to the ground truth. We apply the five feature attribution methods, as discussed in Section 5.1.2, to explain this prediction. The results are shown in Figure 3.

The original explanations, shown in the first row of Figure 3, are roughly located around the sea snake, which is intuitive. The

Table 1: The mstd value of the MeTFA-smoothed RISE and vanilla RISE for Densenet169. The last column is the average mstd among three kinds of $\mathbb{O}_n$ for a fixed $\mathbb{P}$.

$\mathbb{P}\backslash\mathbb{O}_n$	Normal	Uniform	Brightness	Avg
Normal	**0.0591**	**0.0491**	0.0406	**0.0496**
Uniform	0.0613	0.0504	0.0413	0.051
Brightness	0.0668	0.0524	**0.0366**	0.0519
Vanilla RISE	0.1197	0.1219	0.1074	0.1163

second and the third row contains the MeTFA-smoothed and MeTFA-significant maps, generated by applying the two-sided and the one-sided MeTFA to Gradient, IGOS, LIME, RISE and ScoreCAM, respectively. Apparently, the second and third row have better visual quality than the first row. For example, the MeTFA-smoothed Gradient highlights the snake while the original explanation is scattered, and the MeTFA-significant RISE shows that only the snake area is significantly important while the original explanation contains a lot of noises. In general, MeTFA leaves fewer pixels as "undecided" if the original explanation is more stable, *e.g.*, ScoreCAM, and a less stable explanation can benefit more from applying MeTFA, *e.g.*, Gradient. The last two rows show the upper bound maps and lower bound maps. We can directly find which explanations are more stable and which features are uncertain by comparing these two maps. For example, LIME has low stability as the lower bound map and the upper bound map are greatly different while ScoreCAM has high stability as the lower bound map.

5.3 Stability

In this part, we evaluate the stability of MeTFA in the image and text domain. As discussed in Section 3.1, MeTFA samples noises from a distribution $\mathbb{P}$ to estimate the distribution of the explanations. In addition, as discussed in Section 5.1.3, the metric mstd measures the stability of an attribution method by sampling from a noise distribution $\mathbb{O}_n$. In practice, $\mathbb{O}_n$ might be different to $\mathbb{P}$. Therefore, to make the setting more representative for the real applications, we evaluate each combination of $\mathbb{P}$ and $\mathbb{O}$ with several common noises.

5.3.1 Stability in the Image Domain. In the image domain, $\mathbb{P}$ and $\mathbb{O}_n$ are selected from the following three noise distributions which are very common for images: (1) uniform distribution $U(-0.1, 0.1)$, (2) normal distribution $N(0, 0.01)$, (3) brightness, *i.e.*, multiply by a factor n, $n \sim U(0.9, 1.1)$. Although real-world noises may have joint patterns, we apply these perturbations independently to each pixel of the image to simulate the noises introduced by the image sensor [44], *e.g.*, a camera. In addition, neural networks are empirically robust to such random noises [13], thus the correct explanation is probable to remain the same under the random noises, which makes the median value suitable for explaining the original input.

We compare the stability of the MeTFA-smoothed explanation using different $\mathbb{P}$ with the vanilla explanation under different $\mathbb{O}_n$. To compute the mstd, we randomly select 100 images as the test data set D. The results for Densenet169 with RISE algorithm are shown in Table 1. Table 1 shows that every $\mathbb{P}$ can improve the stability under every $\mathbb{O}_n$ when compared to the vanilla RISE, decreasing the mtsd by roughly a half. Therefore, MeTFA does not need to know the "correct" $\mathbb{O}_n$, because the stability transfers across the noise distributions. Extensive experiments on other algorithms and other models are shown in Table 11, Table 12, Table 13 and Table 10 in the Appendix. The results show that MeTFA can significantly increase the stability of LIME, IGOS, Gradient and RISE but has slight effect on ScoreCAM. This may be because LIME and RISE are affected by the sampling process, while Gradient and IGOS are affected by the non-robust features. MeTFA can attenuate the effects of these two factors and thus shows significant increases in the stability. However, ScoreCAM does not involve a sampling process and suffers little from the non-robust features, as shown by a small mtsd for the vanilla explanations. Therefore, ScoreCam benefit less from MeTFA in this sense.

Further, as we can see, the best choice of $\mathbb{P}$ to increase the stability varies for different $\mathbb{O}_n$. For example, when $\mathbb{O}_n$ is Normal, the best $\mathbb{P}$ is Normal; when $\mathbb{O}_n$ is Brightness, the best $\mathbb{P}$ is Brightness. Since the distribution of real-world noise is usually unknown, we take the average among the mstd under three kinds of $\mathbb{O}_n$ for every fixed $\mathbb{P}$ to comprehensively compare the ability of each $\mathbb{P}$ to improve the stability. The results are shown in the last column of Table 1. As we can see, Normal is the best choice for $\mathbb{P}$ to increase the stability under various noises. Extensive experiments on other algorithms and models confirm this conclusion as well, as shown in Table 11, Table 12, Table 13 and Table 10 in the Appendix. Therefore, although normal distribution is not always the best choice for $\mathbb{P}$, it is a good default for applying MeTFA.

Although we have established theoretical results that MeTFA is able to quantify the uncertainty better and converges as fast as SmoothGrad (short for SmoothGrad [39]), their stability under noises on the input is not compared. Therefore, we empirically compare the stability of MeTFA with SG in two settings: (1) there is no noise on the input, which verifies our proof, and (2) there are noises on the input. As we introduced in Section 2, SG samples from the neighborhood of the original image and simply takes the average of all the sampled explanations. However, MeTFA takes the average of the explanations between the lower and upper bound computed from the two-sided MeTFA. This is the only difference between MeTFA-smoothed explanations and the SmoothGrad explanations. For a fair comparison, MeTFA and SG use the same noises sampled from $\mathbb{P}$. Specifically, we randomly sample 100 images from the source dataset as the test data set D to compute mstd. As SG is designed to remove the noise for the Gradient, we take the Gradient as the representative explanation and then compare the stability for MeTFA-smoothed Gradient and SG Gradient.

Since both MeTFA and SG use sampling, their outputs naturally have randomness even if there is no external noise. Thus, we first apply no noise on the input to test the stability of MeTFA and SG, which should verify our proof about the stability advantage of MeTFA. $\mathbb{P}$ is set to be Uniform or Normal. Table 2 shows the ratios of the mstd of the MeTFA-smoothed Gradient over the SG Gradient, *i.e.*, $mstd_{MeTFA-Grad}/mstd_{SG-Grad}$. It can be found that all the ratios are lower than 1, which suggests that the MeTFA-smoothed explanations are more stable than the SG explanations when there is no external noise. Further experiments on the VGG16 model confirms this conclusion, are shown in Table 17 in the appendix. However, this ratio increases when N becomes larger, thus empirically suggests that our asymptotic bound on the convergence is tight, *i.e.*, the lower bound for its convergence rate is $O(1/N)$ as well.

Table 2: The results of $mstd_{MeTFA-Grad}/mstd_{SG-Grad}$ for Resnet50 under two settings: $\mathbb{P}$ is Normal or Uniform and no external noise.

N	Uniform	Normal
10	0.9451	0.9372
30	0.9576	0.9452
50	0.9693	0.9560
70	0.9807	0.9673

Table 3: The results of $mstd_{MeTFA-Grad}/mstd_{SG-Grad}$ for Resnet50 under two settings: $\mathbb{P}$ and $\mathbb{O}_n$ are both Uniform or Normal.

N	Uniform	Normal
10	0.9484	0.9437
30	0.9166	0.9097
50	0.9059	0.8983
70	0.8996	0.8915

Then we test the stability when there are external noises. Specifically, the $\mathbb{O}_n$ and $\mathbb{P}$ are set to be the same and selected from Uniform or Normal. Similar to the noise-free case, we record the ratios of the mstd of the MeTFA-smoothed Gradient over the SG Gradient. Table 3 shows that all the ratios are lower than 1, meaning that MeTFA is always more stable than SG. Moreover, in Appendix A.1.6, we prove that MeTFA only takes the average of $O(\sqrt{N})$ sampled explanations to generate the MeTFA-smoothed explanation. Therefore, the MeTFA-smoothed explanation averages far less sampled explanations than SG (which takes the average of N sampled explanations) to obtain a higher stability because it automatically filters out abnormal extreme values. This property helps MeTFA to be even more stable than SG when the input is noisy.

In conclusion, MeTFA-smoothed explanations are more suitable when the vanilla explanations are vulnerable to the effect of non-robust features (e.g., Gradient, IGOS) or the sampling process (e.g., RISE, LIME).

5.3.2 Stability in the Text Domain. In the text domain, we use synonym substitution as noise, because different words express similar meanings in a sentence. Formally, this noise $\mathbf{P}(p)$ replaces every word by its synonym independently with probability p. In this experiment, we set both $\mathbb{P}$ and $\mathbb{O}_n$ to $\mathbf{P}(0.5)$. Specifically, we use wordnet in nltk [7] for synonym substitution and do not require the predicted class to keep the same in the experiment, as we want to simulate the noise in the real world. As discussed in Section 5.1, we use LEMNA as the target explanation and a bidirectional LSTM as the target model. Similar to Section 5.3.1, We randomly select 100 toxic texts whose number of words are between 40 and 80 as the test data set D.

The result is shown in Table 4, where the number of samples of LEMNA is 2000. It can be found that the mstd value of MeTFA-smoothed LEMNA is significantly smaller than that of the vanilla LEMNA, which means that MeTFA can increase the stability of LEMNA as well. The results for the Toxic Commnet, where the number of samples of LEMNA is 500, are shown in the appendix (Table 19), and the conclusion is the same, i.e., MeTFA can increase the stability of LEMNA.

Table 4: The mstd values for LEMNA and MeTFA-smoothed LEMNA when $\mathbb{P}$ and $\mathbb{O}_n$ are both $\mathrm{P}(0.5)$.

Dataset	LEMNA	MeTFA-smoothed LEMNA
Toxic Comment	0.1803	**0.0891**
IMDb Reviews	0.3012	**0.1691**

Table 5: The results with the faithfulness metrics. The tuple in the table is structured as (the score of the MeTFA-smoothed explanation, the score of the vanilla explanation).

method	insertion	deletion	overall
ScoreCAM	(0.5897,**0.6101**)	(0.1571,**0.1439**)	(0.4326,**0.4662**)
RISE	(0.5508,**0.5556**)	(0.1767,**0.1550**)	(0.3741,**0.4006**)
IGOS	(**0.3881**,0.3360)	(0.1107,**0.1002**)	(**0.2774**,0.2358)

Table 6: The results with the robust faithfulness metric. $\mathbb{O}_n = N(0, 0.1)$. The structure of the table is similar to Table 5.

method	RI	RD	RO
ScoreCAM	(**1.2915**,0.9429)	(0.4279,**0.4039**)	(**0.8636**,0.5390)
RISE	(**1.2959**,0.8626)	(**0.4497**,0.4846)	(**0.8462**,0.3780)
IGOS	(**0.5061**,0.4152)	(0.2249,**0.1640**)	(**0.2812**,0.2512)

5.4 Faithfulness

Faithfulness to the model is an essential property for an explanation. The MeTFA-smoothed explanation approximates the median of the attribution maps under some $\mathbb{P}$. The following experiments shows that MeTFA greatly increases the robust faithfulness while maintaining the faithfulness level. Therefore, MeTFA-smoothed explanations find more robust features used by the model.

5.4.1 Faithfulness in the Image Domain. In the image domain, we use insertion, deletion, overall, RI, RD and RO to estimate the faithfulness of an explanation which are used to measure whether the features highlighted by an attribution map support a model's prediction. However, the gradient-based explanations are not designed to highlight the supportive features, and thus these metrics are not suitable for them. Moreover, these metrics can only evaluate the continuous attribution maps while LIME generates a discrete (in fact, a binary) map. Therefore, we do not test these metrics for Gradient, LRP and LIME and only test them for IGOS, ScoreCAM and RISE. The result is evaluated on the VGG16 model, and the metrics are averaged on 1000 randomly chosen images. In this experiment, $\mathbb{P}$ is set to be Uniform, and $\mathbb{O}_n$ is chosen from Uniform and Normal.

For the vanilla insertion, deletion and overall, the average scores of the 1000 test images are shown in Table 5. The bold digits in the table represent the higher faithfulness. It shows that, for ScoreCAM and RISE, two-sided MeTFA slightly decreases the value of insertion and increases the value of deletion, which suggests that two-sided MeTFA slightly reduces the faithfulness of the vanilla explanation algorithms. For IGOS, two-sided MeTFA slightly increases the value of insertion and increases the value of deletion. Thus, for such contradictory introduced in Section 5.1, overall is used to evaluate the faithfulness, and the results show that the two-sided MeTFA increases the faithfulness of IGOS. In general, the two-sided MeTFA maintains the faithfulness because the overall score is similar to the vanilla explanation.

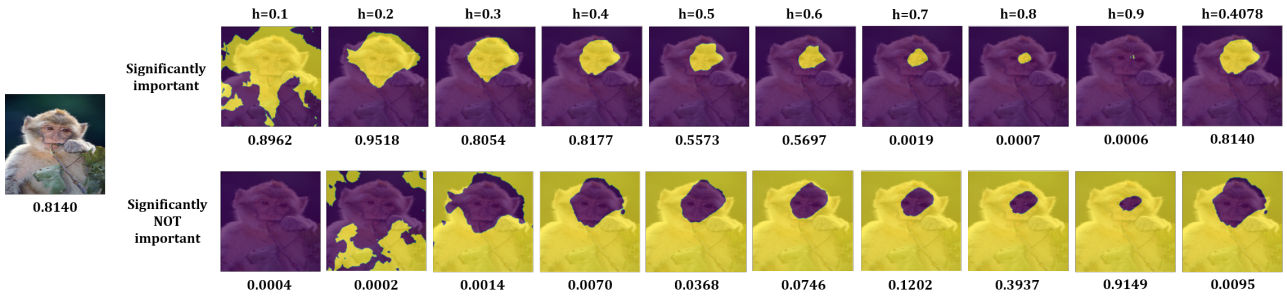

Figure 4: The MeTFA-significant RISE with h from 0.1 to 0.9. The last column is the MeTFA-significant explanations with the recommended threshold h. The values under the images represent the predicted score of the original label, *i.e.*, macaque, for the significantly important areas.

For RI, RD and RO, the average scores of the 1000 test images are shown in Table 6 where $\mathbb{O}_n = N(0, 1)$. As we can see, the two-sided MeTFA significantly increases the robust faithfulness of the three vanilla explanations. Further experiments that calculate the RI, RD and RO with $\mathbb{O}_n = U(-0.1, 0.1)$ confirm this conclusion and the results are shown in Table 15 in the appendix. All of these results show that MeTFA significantly increases the robust faithfulness for the three explanations, regardless of $\mathbb{O}_n$ and $\mathbb{P}$ are the same or not.

As we can see, the vanilla RISE and ScoreCAM have higher overall score than MeTFA-smoothed ones. The reason could be that the explanations without MeTFA overfit the non-robust features or artifacts [14], and thus receiving a higher faithfulness value, just as some models have higher accuracy on clean images but are more vulnerable to noises. MeTFA eliminates the effect of some non-robust features due to the sampling and the test. Thus, MeTFA slightly decreases the faithfulness of some explanation methods using the traditional metrics, but significantly increases the faithfulness using the proposed robust metrics.

5.4.2 Faithfulness in the Text Domain. In the text domain, we use FDT, FAT, ST and their corresponding robust metrics to estimate the faithfulness of an explanation. Similar to Section 5.3, we use synonym substitution to generate noise and set $\mathbb{P}$ and $\mathbb{O}_n$ to be $\mathbf{P}(0.5)$. As introduced in Section 5.1, the value of FDT changes with the number of processed features, *i.e.*, n. The results of FDT, FAT and ST with different n are shown in Figure 5 for Toxic Comment dataset, where the number of samples of LEMNA is 2000. It can be found that the FDT value (lower is better) of the MeTFA-smoothed LEMNA is always lower while the other two values (higher is better) are always higher, which suggests that the MeTFA-smoothed LEMNA is more faithful. The results of RFDT, RFAT and RST with different n are shown in Figure 6 for Toxic Comment dataset, where the number of samples of LEMNA is 2000. The results confirm that the two-sided MeTFA increases the faithfulness of LEMNA. Further, we change the strength of the noise by setting $\mathbb{O}_n$ to $\mathbf{P}(0.3)$ and $\mathbf{P}(0.7)$. The results of the three robust faithfulness metrics are shown in Figure 11 and Figure 12 in the appendix, respectively, which consistently shows that MeTFA is better. Moreover, the results with another dataset (i.e., IMDb Reviews) and another number of samples for LEMNA (i.e., 500) are shown in Figure 13 and Figure 14 in the appendix, respectively. All of these results show that MeTFA can increase LEMNA's faithfulness regardless of the existence of the real-world noises and the strength of the random noises.

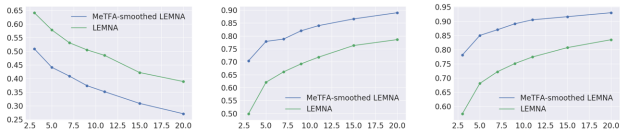

Figure 5: The results of FDT, FAT and ST with different n when $\mathbb{O}_n$ is $\mathrm{P}(0.5)$. The results of FDT, FAT and ST are shown from left to right, respectively.

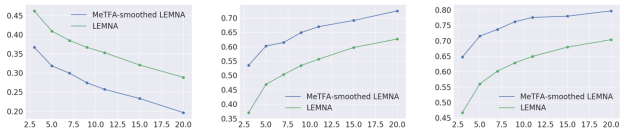

Figure 6: The results of of RFDT, RFAT and RST with different n when $\mathbb{O}_n$ is $\mathrm{P}(0.5)$. The results of RFDT, RFAT and RST are shown from left to right, respectively.

5.5 Impact of the Key Parameters

In this part, we discuss the impact of several key parameters on the capabilities of MeTFA, including the threshold h of the MeTFA-significant map, the number of sampled explanations N and the confidence level α.

5.5.1 Threshold h. Although we recommend to determine the threshold h of the MeTFA-significant map by finding the optimal break, as discussed in Section 4.1, h is still a customizable parameter. In this part, we illustrate how h influences the MeTFA-significant explanation and then show the advantage of applying the recommended threshold.

Figure 4 shows the MeTFA-significant maps for RISE on an image of a macaque. The first row highlights the significantly important features, and the second row highlights the significantly unimportant features. When h increases, the significantly important area becomes smaller, and the significantly unimportant area becomes larger, which is intuitive. To understand how well the highlighted area represents the model's prediction, we black out the image except the significantly important area and record the model's predicted score of the original label. When $h = 0.1$ and 0.2, the significantly important region filters out the noisy features, leading to a higher predicted score compared to the original image. When $h = 0.3$ and 0.4, the significantly important map keeps the predicted score with a smaller region. When $h = 0.5$ and 0.6, the area of the significantly important map is further reduced and the predicted score drops, but

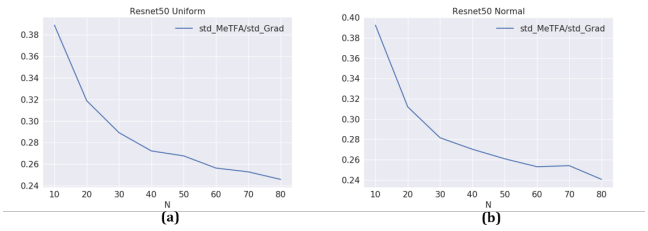

Figure 7: The results of $mstd_{MeTFA-Grad}/mstd_{Grad}$ for Resnet-50 when N varies from 10 to 80.

the prediction remains the same as the score is still larger than 0.5. Finally, when $h \geq 0.7$, the significantly important region is too small to keep enough information which causes a quick decrease of the predicted score. These phenomena show that the significantly important map correctly points out the features supporting the prediction of the model. In addition, when $h = 0.8$, the significantly unimportant map covers almost the whole image but still gets a low score, suggesting that the significantly unimportant map correctly points out the features which do not support the prediction of the model.

By applying the optimal break method discussed in Section 4.1, the recommended h for this example is 0.4078. Using this threshold, the significantly important map gets a high score with a small area. In addition, this threshold is almost the same to the score-area margin, $h = 0.4$, as a smaller threshold keeps a much larger area and a higher threshold gets a much smaller predicted score. This example shows that the recommended method of determining the threshold is good for usage. Therefore, in the following applications (Section 6.1), we use the recommended way to determine h for the MeTFA-significant map.

5.5.2 Number of Sampled Explanations N. Although we give the lower bound for N to achieve a confidence level α in Section 4.3, N is a customizable parameter as long as it is greater than the lower bound. In this part, we demonstrate how N influences the stability of the MeTFA-smoothed explanation. As discussed in Section 5.3, we set the $\mathbb{P}$ and $\mathbb{O}_n$ to be the same and experiment with Normal and Uniform distributions. The results for the ResNet-50 are shown in Figure 7. As expected, the mstd of MeTFA-smoothed Gradient decreases when N increases, meaning that the explanation is more stable with a larger N. Therefore, a user can custom N according to the trade-off between the stability of the explanation and the tolerance of computational costs. However, even a small N, *e.g.*, $N = 10$, can bring significant stability benefits, as the std is reduced by over a half.

5.5.3 Confidence Level α. α is a core parameter for hypothesis testing, and a lower α causes a more stringent test result. As discussed in Section 4.3, a lower α requires a higher N. However, with a fixed large N, whether the choice of α makes a significant difference to the stability of the explanation remains unknown. To answer this practical question, we fix $N = 50$ and experiment with different α. Similar to the discussion of N, we test the stability of the MeTFA-smoothed Gradient for Resnet-50 by setting $\mathbb{P}$ and $\mathbb{O}_n$ both to be Normal and Uniform, respectively. The results are shown in Table 7. It shows that the stability does not change much with a smaller α when we reduce α from 0.05 to 0.0001. Therefore, MeTFA is insensitive to the value of α. Further experiments on Densenet-169 imply

Table 7: The results of $mstd_{MeTFA-Grad}/mstd_{Grad}$ for Resnet50 under two settings: $\mathbb{P}$ and $\mathbb{O}_n$ are both Normal or Uniform.

α	Uniform	Normal
0.05	0.2558	0.2609
0.01	0.2553	0.2607
0.005	0.2552	0.2608
0.001	0.2552	0.2610
0.0005	0.2554	0.2613
0.0001	0.2557	0.2619

the same conclusion, as shown in Table 18 in the appendix. The intuition of this result is that $\alpha = 0.05$ already implies the probability of noises to be tested significant is very small, and thus reducing it further does not benefit as much. Therefore, we set $\alpha = 0.05$ in the following application, which is common for a hypothesis test.

6 APPLICATION

In this section, we apply MeTFA to two applications closely related to security: detecting context bias in semantic segmentation and defending adversarial examples against the explanation-oriented attack. In this section, $\mathbb{P}$ is set to $U(-0.1, 0.1)$.

6.1 Context Bias Detection

As a component of the autonomous driving, semantic segmentation is of great importance. Formally, suppose that the class to be segmented is c, *e.g.*, rider, and I is the input image. A semantic segmentation model F_c basically predicts for each pixel whether it belongs to c, represented by a probability score, and the segmentation result, denoted by R, is the set of all pixels that are predicted to be c. An explanation for the segmentation R is an attribution map that highlights the area R that the segmentation is based on. The highlighted area may include additional information that supports the prediction, *i.e.*, $M \setminus R$. For example, it may highlight a bike that supports the segmentation of a rider, but is not included in the segmentation for the rider. This additional areas are called context bias for class c. Similar to the image classification task, an explanation for the segmentation is faithful if only keeping the highlighted areas is sufficient to produce the correct segmentation for c. Formally, we define the faithfulness value as follows:

$$m_f = 1 - \frac{Sum(Relu(R \odot (F_c(I) - F_c((R \cup M) \odot I))))}{Sum(R)}$$

Basically, this metric measures how much the segmentation scores drop if we use only the explanation combined with the segmentation area to produce a new segmentation. A high m_f means the model produces similar segmentation with the only area highlighted by the explanation. Similarly, we measure the robust faithfulness under noises sampled from O_n, defined by $E_{n \sim O_n} m_f(I + n)$.

GridSaliency [20] is the SOTA explanation algorithm for semantic segmentation. An example is shown in Figure 8 (a) and (b). Figure 8 (a) is the result of the model segmenting the rider class, and Figure 8 (b) highlights the context bias of the rider class using GridSaliency. This result means that the model needs both the rider (Figure 8 (a)) and the bike (Figure 8 (b)) to recognize the rider. Using GridSaliency, engineers can debug a model when the model relies on the wrong context bias. However, the explanation of GridSaliency is vulnerable to random noise, which may mislead the practice. For example, after

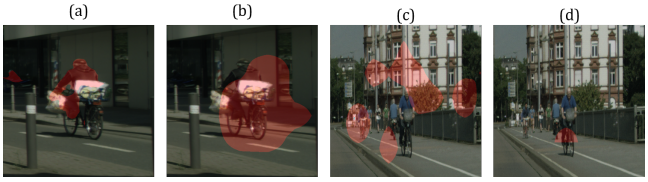

Figure 8: An example of the explanation for the semantic segmentation model. (a) and (b) are generated with the clean image. (a) is the *rider* segmentation result and (b) is its explanation generated by GridSaliency. (c) and (d) are the explanation of the rider class with a noisy image. (c) is the original GridSaliency map, while (d) is the MeTFA-significant map.

adding some noises to the original image, the attribution map for the rider becomes irrelevent, as shown in Figure 8 (c).

We apply the MeTFA-significant map to fix this issue. As shown in Figure 8 (d), the MeTFA-smoothed map correctly highlights the bike again. We further evaluate the faithfulness of the MeTFA-significant explanations using the popular semantic segmentation dataset CityScapes [9] as the source dataset and the pre-trained PSPNet with R-50-D8 backbone [8] as the target model . We select three classes, tree, rider and car, as the target classes because intuitively rider has a strong context bias (bike) while trees and cars do not. For each class, we randomly select 100 test images where the segmentation size is larger than 600 pixels, to ensure that there exists at least one object segmented by the model as c rather than some misclassified noisy pixels. To test the faithfulness of an explanation method in a noisy environment, we use noises sampled from $U(-0.1, 0.1)$ to compute the robust m_f. For a fair comparison, we apply the same h for the vanilla GridSaliency as well, *i.e.*, we take the set of pixels with score higher than h as the explanation of the vanilla GridSaliency. The results are shown in Table 8. It can be seen that the MeTFA-significant explanations highlight far less pixels, e.g., about 2% for trees, to maintain 99%+ faithfulness, which suggests that the one-sided MeTFA filters out many noisy pixels in the vanilla GridSaliency map and keeps the pixels that the model really relies on. This can help engineers confidently determine if a model has context bias by looking at the the region the model significantly relies on.

Furthermore, from Table 8, we can see that MeTFA filters out most of the context biases for the class tree and car, at 98% and 92%, respectively, while maintaining 40% of the context bias for the class rider. This is intuitive because the model needs the context bias, *e.g.*, bike, to determine whether a person is a rider. However, for trees and cars, the model does not need context bias to do so. Therefore, this results suggest that MeTFA is good at removing the false positives for the target classes and keeping only the correct context biases.

In conclusion, the MeTFA-significant map can remove the noise of the attribution map and point out the context bias more accurately and confidently.

6.2 Defending Explanation-Oriented Attacks

Explanations are designed to help human understand and trust the model. However, recent works show that the explanation can be manipulated. Manipulation attack [11] is able to keep the model's prediction unchanged, but manipulate the attribution maps generated by the explanation algorithms arbitrarily. As shown in the second row

Table 8: Evaluation of the highlighted area and the faithfulness of the attribution map for the segmentation model PSPNet. The second column is the faithfulness ratio of the MeTFA-significant map divided over the GridSaliency map. The third column is the ratio of the highlighted area.

classes	m_{MeTFA}/m_{GS}	$\|M_{MeTFA}\|/\|M_{GS}\|$
tree	0.9983	0.0198
rider	0.9904	0.3955
car	0.9915	0.080

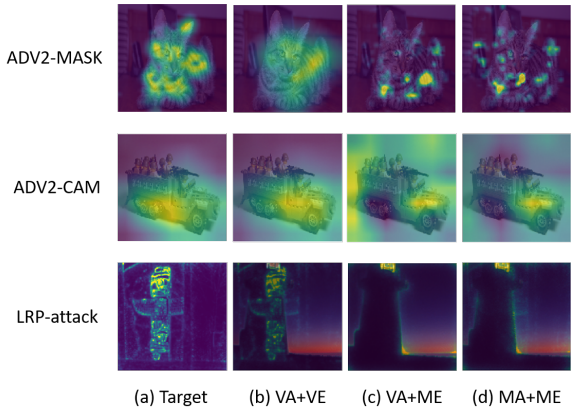

Figure 9: Examples of manipulation attack for MASK, CAM and LRP attack. (a) are the target maps. The attacker uses the vanilla attack (VA) to generate an adversarial example. Then the defender uses the vanilla explanation (VE) to generate a map for the adversarial example (b) or use the MeTFA-smoothed explanation (ME) to generate a map for the adversarial example (c). The attacker can also use adaptive attack against MeTFA (MA) to generate an adversarial example. Then the defender uses the MeTFA-smoothed explanation (ME) to generate a map for the adversarial example (d).

of Figure 9 (a) and (b), an attacker can manipulate the explanation to be similar to a target map. Moreover, some AI systems use feature attribution to detect adversarial explanations, but ADV^2 [50] can evade such detection by manipulating the adversarial example's explanation to be similar to the benign one. As shown in the first row of Figure 9 (a) and (b), an attacker changes the predicted label of the image from check to sandbar while manipulating its explanation (b) to be similar to the benign one (a). ADV^2 attack can be decomposed into two steps: an attacker first attacks the prediction of the model only and then manipulates the explanation to the benign one while maintaining the target label. Therefore, the core of the above two attacks is the same, i.e., manipulate the explanation to a target map while maintaining the predicted label.

We conduct experiments to test MeTFA's ability to defend the explanation-oriented attack quantitatively. Similar to Section 5.1.1, we use ILSVRC2012 val as the source dataset. We test the attack on MeTFA-smoothed CAM [50] (a CAM -based explanation), MeTFA-smoothed LRP[11] (a gradient-based explanation) and MeTFA-smoothed MASK [50] (an optimization-based explanation). Since the existing attack methods rely on the gradient relation between the attribution map and the input image, they could not attack the

Table 9: The distance between the maps for CAM, LRP and MASK.

	$d(m_1, m_2)$	$d(m_1, m_3)$	$d(m_1, m_4)$
CAM	0.0934	0.2785	0.1705
LRP	0.0340	0.0657	0.0635
MASK	0.1453	0.1667	0.1648

sample-based explanations, where no gradient information is available. All the attack methods follow the default settings of the original paper.

The aim of the attacker is to manipulate the explanation to a target pattern while keeping the predicted label. Correspondingly, the aim of the defender is to make the generated explanation different from the target pattern. We evaluate the defense capabilities of MeTFA from two aspects: visual effects and quantitative analysis. For each aspect, we conduct experiments for both conditions where the attacker knows or does not know MeTFA. Formally, the objective functions to optimize the adversarial example for the original attack (Equation 1) and the adaptive attack (Equation 2) are as follows:

$$x = \arg\min_x(\lambda_1\|f_c(x) - f_c(x_0)\| + \lambda_2\|x - x_0\| + \lambda_3\|E(f_c, x) - E_t\|) \tag{1}$$

$$x = \arg\min_x(\lambda_1\|f_c(x) - f_c(x_0)\| + \lambda_2\|x - x_0\| + \lambda_3\|E(f_c, x) - E_t\| + \lambda_4\|E_M(f_c, x) - E_t\|) \tag{2}$$

where $f_c(x)$ returns the predicted score of class c, x_0 is the original data, $E(f_c, x)$ returns the vanilla attribution map of x, E_t is the target attribution map and $E_M(f_c, x)$ returns the MeTFA-Smoothed attribution map of x. In the experiment, we consider the strongest adaptive attacker, i.e., the attacker uses the hyperparameters exactly the same as the defender who uses MeTFA to denfend against the adversarial examples.

First, the visual results of MeTFA are shown in Figure 9 (c) and (d) for MeTFA-smoothed explanation of the vanilla adversarial examples and that of the adaptive adversarial examples. As we can see, whether the attacker knows MeTFA, the MeTFA-smoothed explanations are very different to the targets, which suggests the ability of MeTFA to defend against adversarial examples in practice.

Second, we quantify the ability of the MeTFA to defend against the attack. Formally, we denote the target map as m_1 (e.g., Figure 9(a)), the map generated with vanilla explanation for the vanilla adversarial example as m_2 (e.g., Figure 9(b)), the map generated with MeTFA-smoothed explanation for the vanilla adversarial example as m_3 (e.g., Figure 9(c)) and the map generated with MeTFA-smoothed explanation for the adaptive adversarial example as m_4 (e.g., Figure 9(d)). We define the distance between two maps as $d(m, n) = \frac{1}{|S|}\sum_{(i,j)\in S}|m_{ij} - n_{ij}|$, where m_{ij} is the value of the pixel (i, j) in m. To quantify the difference between these maps, we test 100 random selected images from ImageNet and show the average distance of the 100 images in Table 9. When the attacker is not adaptive, we can see that $d(m_1, m_3)$ is significantly greater than $d(m_1, m_2)$, which suggest MeTFA can defend against such attack. When the attacker is adaptive for MeTFA, we can see $d(m_1, m_4)$ is still significantly greater than $d(m_1, m_2)$. Therefore, MeTFA can weaken the attacker's ability to manipulate the explanation even when the attacker knows MeTFA.

7 DISCUSSION

Bonferoni correction. Bonferoni correction guides researchers to use union bounds when computing confidence interval. We do not do a Bonferoni correction for computing confidence intervals over multiple features. Instead, MeTFA define a two-fold hypothesis testing to generate confidence interval for each feature individually. This is because Bonferoni correction requires a lot more queries for high dimensional data (e.g., a 224×224-dimensional image) to get tighter bounds and thus is inefficient in practice. For efficiency, we consider the confidence interval for each feature individually and the experiments show it works well.

Asymptotic Quantification from Extending SmoothGrad. Although SmoothGrad does not quantify the uncertainty of explanation, it can be extended to provide an asymptotic confidence bound. Using Jackknife method [45], one can estimate the standard deviation σ of the smoothed explanation μ. By the central limit theorem [43], the smoothed explanation asymptotically converges to a normal distribution $\mathcal{N}(\mu, \sigma^2)$. Therefore, the α confidence bound is $[\mu - p_{\alpha/2}\sigma, \mu + p_{\alpha/2}\sigma]$, where $p_{\alpha/2}$ is the upper $\alpha/2$ quantile of the standard normal distribution. However, this bound is only valid when N is large, while our bound is valid for all N.

Limitation of MeTFA. As illustrated in Section 5.4, if an explanation algorithm produces unfaithful explanations due to the effectiveness of non-robust features or the randomness in the algorithm, then MeTFA can significantly help and produce better explanation. However, if the explanation algorithm has major problems, such as attributing all features with the same value, then MeTFA can hardly generate faithful explanations. In other words, MeTFA can only make weak explanations stronger, by removing noises in the attribution maps.

8 CONCLUSION

In this paper, we propose MeTFA, the first work to quantify and reduce the randomness in feature attribution methods with theoretical guarantees. By evaluating with extensive experiments, we show that MeTFA can increase the stability of explanation while maintaining the faithfulness. With the proposed robust faithfulness metrics, we show that MeTFA-smoothed explanations significantly increase the explanation's ability to locate robust features. In addition, we demonstrate that MeTFA can detect context bias in the semantic segmentation model more accurately and defend against the explanation-oriented attack, which shows its great potential in practice.

9 ACKNOWLEDGMENTS

We would like to gratefully thank the anonymous reviewers for their helpful feedback. This work was partly supported by the Zhejiang Provincial Natural Science Foundation for Distinguished Young Scholars under No. LR19F020003, NSFC under No. 62102360, U1936215, and U1836202, and the Open Research Projects of Zhejiang Lab under No. 2022RC0AB01. Ting Wang is partially supported by the National Science Foundation under No. 1951729, 1953893, 2119331, and 2212323.

REFERENCES

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. 2018. Sanity Checks for Saliency Maps. In *Advances in Neural*

Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (Eds.). 9525–9536. https://proceedings.neurips.cc/paper/2018/hash/294a8ed24b1ad22ec2e7efea049b8737-Abstract.html

[2] Sushant Agarwal, Shahin Jabbari, Chirag Agarwal, Sohini Upadhyay, Steven Wu, and Himabindu Lakkaraju. 2021. Towards the unification and robustness of perturbation and gradient based explanations. In *International Conference on Machine Learning*. PMLR, 110–119.

[3] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. Drebin: Effective and explainable detection of android malware in your pocket.. In *Ndss*, Vol. 14. 23–26.

[4] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one* 10, 7 (2015), e0130140.

[5] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. 2010. How to explain individual classification decisions. *The Journal of Machine Learning Research* 11 (2010), 1803–1831.

[6] Jenny A. Baglivo. 2005. *Mathematica laboratories for Mathematical Statistics: Emphasizing simulation and computer intensive methods*. Society for Industrial and Applied Mathematics.

[7] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

[8] MMSegmentation Contributors. 2020. MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark. https://github.com/open-mmlab/mmsegmentation.

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. 2016. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[10] Bao Gia Doan, Ehsan Abbasnejad, and Damith C Ranasinghe. 2020. Februus: Input purification defense against trojan attacks on deep neural network systems. In *Annual Computer Security Applications Conference*. 897–912.

[11] Ann-Kathrin Dombrowski, Maximilian Alber, Christopher J Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. 2019. Explanations can be manipulated and geometry is to blame. *arXiv preprint arXiv:1906.07983* (2019).

[12] Tianyu Du, Shouling Ji, Lujia Shen, Yao Zhang, Jinfeng Li, Jie Shi, Chengfang Fang, Jianwei Yin, Raheem Beyah, and Ting Wang. 2021. Cert-RNN: Towards Certifying the Robustness of Recurrent Neural Networks.. In *CCS*. 516–534.

[13] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. 2016. Robustness of Classifiers: From Adversarial to Random Noise. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Barcelona, Spain) *(NIPS'16)*. Curran Associates Inc., Red Hook, NY, USA, 1632–1640.

[14] Ruth C Fong and Andrea Vedaldi. 2017. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*. 3429–3437.

[15] Chong Fu, Xuhong Zhang, Shouling Ji, Jinyin Chen, Jingzheng Wu, Shanqing Guo, Jun Zhou, Alex X Liu, and Ting Wang. 2022. Label inference attacks against vertical federated learning. In *31st USENIX Security Symposium (USENIX Security 22), Boston, MA*.

[16] Amirata Ghorbani, Abubakar Abid, and James Y. Zou. 2019. Interpretation of Neural Networks Is Fragile. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 3681–3688. https://doi.org/10.1609/aaai.v33i01.33013681

[17] Jinjin Gu and Chao Dong. 2021. Interpreting Super-Resolution Networks With Local Attribution Maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9199–9208.

[18] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. 2018. Lemna: Explaining deep learning based security applications. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 364–379.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[20] Lukas Hoyer, Mauricio Munoz, Prateek Katiyar, Anna Khoreva, and Volker Fischer. 2019. Grid saliency for context explanations of semantic segmentation. *arXiv preprint arXiv:1907.13054* (2019).

[21] G. Jenks. 1967. The Data Model Concept in Statistical Mapping.

[22] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T. Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2019. The (Un)reliability of Saliency Methods. In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller (Eds.). Lecture Notes in Computer Science, Vol. 11700. Springer, 267–280. https://doi.org/10.1007/978-3-030-28954-6_14

[23] Jungbeom Lee, Jihun Yi, Chaehun Shin, and Sungroh Yoon. 2021. BBAM: Bounding Box Attribution Map for Weakly Supervised Semantic and Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2643–2652.

[24] Changjiang Li, Li Wang, Shouling Ji, Xuhong Zhang, Zhaohan Xi, Shanqing Guo, and Ting Wang. 2022. Seeing is living? rethinking the security of facial liveness verification in the deepfake era. *CoRR abs/2202.10673* (2022).

[25] Jinfeng Li, Tianyu Du, Shouling Ji, Rong Zhang, Quan Lu, Min Yang, and Ting Wang. 2020. {TextShield}: Robust Text Classification Based on Multimodal Embedding and Neural Machine Translation. In *29th USENIX Security Symposium (USENIX Security 20)*. 1381–1398.

[26] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Portland, Oregon, USA, 142–150. http://www.aclweb.org/anthology/P11-1015

[27] Yuhao Mao, Chong Fu, Saizhuo Wang, Shouling Ji, Xuhong Zhang, Zhenguang Liu, Jun Zhou, Alex X Liu, Raheem Beyah, and Ting Wang. 2022. Transfer Attacks Revisited: A Large-Scale Empirical Study in Real Computer Vision Settings. *arXiv preprint arXiv:2204.04063* (2022).

[28] Ren Pang, Zhaohan Xi, Shouling Ji, Xiapu Luo, and Ting Wang. 2022. On the Security Risks of {AutoML}. In *31st USENIX Security Symposium (USENIX Security 22)*. 3953–3970.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

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

[31] Vitali Petsiuk, Rajiv Jain, Varun Manjunatha, Vlad I. Morariu, Ashutosh Mehra, Vicente Ordonez, and Kate Saenko. 2021. Black-Box Explanation of Object Detectors via Saliency Maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11443–11452.

[32] Vitali Petsiuk, Rajiv Jain, Varun Manjunatha, Vlad I Morariu, Ashutosh Mehra, Vicente Ordonez, and Kate Saenko. 2021. Black-box explanation of object detectors via saliency maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11443–11452.

[33] Zhongang Qi, Saeed Khorram, and Fuxin Li. 2019. Visualizing Deep Networks by Optimizing with Integrated Gradients.. In *CVPR Workshops*, Vol. 2.

[34] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 1135–1144.

[35] Marko Robnik-Šikonja and Marko Bohanec. 2018. *Perturbation-Based Explanations of Prediction Models*. Springer International Publishing, Cham, 159–175. https://doi.org/10.1007/978-3-319-90403-0_9

[36] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* 115, 3 (2015), 211–252. https://doi.org/10.1007/s11263-015-0816-y

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

[38] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *In Workshop at International Conference on Learning Representations*.

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

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[41] Jorg Wagner, Jan Mathias Kohler, Tobias Gindele, Leon Hetzel, Jakob Thaddaus Wiedemer, and Sven Behnke. 2019. Interpretable and fine-grained visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9097–9107.

[42] Haofan Wang, Zifan Wang, Mengnan Du, Fan Yang, Zijian Zhang, Sirui Ding, Piotr Mardziel, and Xia Hu. 2020. Score-CAM: Score-weighted visual explanations for convolutional neural networks. In *Proceedings of the IEEE/CVF conference*

on computer vision and pattern recognition workshops. 24–25.

[43] Wikipedia. 2022. Central limit theorem — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Central%20limit%20theorem&oldid=1080658839. [Online; accessed 03-April-2022].

[44] Wikipedia. 2022. Image noise — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Image%20noise&oldid=1080156929. [Online; accessed 07-April-2022].

[45] Wikipedia. 2022. Jackknife resampling — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Jackknife%20resampling&oldid=1076175481. [Online; accessed 03-April-2022].

[46] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. 2019. On the (In)fidelity and Sensitivity of Explanations. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (Eds.). 10965–10976. https://proceedings.neurips.cc/paper/2019/hash/a7471fdc77b3435276507cc8f2dc2569-Abstract.html

[47] Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Suggala, David I Inouye, and Pradeep K Ravikumar. 2019. On the (in) fidelity and sensitivity of explanations. *Advances in Neural Information Processing Systems* 32 (2019), 10967–10978.

[48] Qinglong Zhang, Lu Rao, and Yubin Yang. 2021. Group-CAM: Group Score-Weighted Visual Explanations for Deep Convolutional Networks. *arXiv preprint arXiv:2103.13859* (2021).

[49] Qinglong Zhang, Lu Rao, and Yubin Yang. 2021. A Novel Visual Interpretability for Deep Neural Networks by Optimizing Activation Maps with Perturbation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3377–3384.

[50] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. 2020. Interpretable deep learning under fire. In *29th {USENIX} Security Symposium ({USENIX} Security 20)*.

[51] Haibin Zheng, Zhiqing Chen, Tianyu Du, Xuhong Zhang, Yao Cheng, Shouling Ti, Jingyi Wang, Yue Yu, and Jinyin Chen. 2022. NeuronFair: interpretable white-box fairness testing through biased neuron identification. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)*. IEEE, 1519–1531.

[52] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. 2016. Learning deep features for discriminative localization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2921–2929.

[53] Kai Zhu and Tao Zhang. 2021. Deep reinforcement learning based mobile robot navigation: A review. *Tsinghua Science and Technology* 26, 5 (2021), 674–691. https://doi.org/10.26599/TST.2021.9010012

A APPENDIX

A.1 Proofs

A.1.1 Proof of Proposition 1. We only prove the case when $h \geq V$. The other case can be derived similarly. Since $ct_j(h) := \sum_{i=1}^{N} I(e_{ij} \geq h)$ is the sum of i.i.d. Bernoulli variables, for a fixed $q_j(h) := P(e_{ij} \geq h)$ we can easily obtain $P(ct_j(h) \geq m)$. The problem here is that we do not know $q_j(h)$, except that $q_j(h) \geq 0.5$ for $h \geq V$. Therefore, we can get the upper bound of $P(ct_j(h) \geq m)$ via $\max_{q_j(h) \geq 0.5} P(ct_j(h) \geq m)$.

We have known that $ct_j(h) \sim B(p)$, where $p = q_j(h) \leq q_j(V) = 0.5$. Therefore, $P(ct_j(h) = m) = \binom{N}{m} \times p^m \times (1-p)^{N-m}$. For a fixed m, the probability increases monotonically for $p \in [0, m/N)$ and decreases monotonically for $p \in (m/N, 1]$. Thus the probability has and only has one maximal, achieved by $p_* = \min(0.5, m/N)$. Therefore, $P(ct_j(h) = m) \leq \binom{N}{m} \times p_*^m \times (1-p_*)^{N-m}$ for every m. Adding up this inequality w.r.t. m proves the statement.

A.1.2 Proof of Theorem 1. For $H_0 : V_j \leq h$, the p-value is the probability of abnormal event $P(ct_j(h) \geq m)$ with a large m. Therefore, according to Proposition 1, its p-value is less than or equal to $\sum_{i=k}^{N} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$. Similarly, for $H_0 : V_j \geq h$, the p-value is the probability of abnormal event $P(ct_j(h) \leq m)$ with a small m. Therefore, according to Proposition 1, its p-value is less than or equal to $\sum_{i=0}^{k} \binom{N}{i} \times p_*^i \times (1-p_*)^{N-i}$.

A.1.3 Proof of Proposition 2. Since $q_j(h) = 0.5$ and $ct_j(h) \sim B(q_j(h))$, we have $P(ct_j(h) \leq m_1) \leq \sum_{i=0}^{m_1} \binom{N}{i} \times 0.5^N$ and $P(ct_j(h) \geq m_2) \leq \sum_{i=m_2}^{N} \binom{N}{i} \times 0.5^N$. Adding them together completes the proof.

A.1.4 Proof of Theorem 2. For $H_0 : V_j = h$, the p-value is the probability of abnormal event $P(ct_j(h) \geq m)$ with a large m and $P(ct_j(h) \leq m)$ with a small m. Therefore, according to Proposition 2, its p-value is less than or equal to $\sum_{i \in \{0,\ldots,k_1\} \cup \{k_2,\ldots,N\}} 0.5^N \times \binom{N}{i}$.

A.1.5 Proof of Theorem 3. By Proposition 2, under $H_0 : V_j = h_0$, $P_{h_0}(ct_j(h_0) \in (k_1, k_2)) = 1 - P(ct_j(h_0) \leq k_1) - P(ct_j(h_0) \geq k_2) = 1 - 2 \times P(ct_j(h_0) \leq k_1) \geq 1 - 2 \times \sum_{i=0}^{k_1} 0.5^N \times \binom{N}{i} \geq 1 - \alpha$. By the arbitrariness of h_0, we can write $P_h(ct_j(h) \in (k_1, k_2)) \geq 1 - \alpha$ for any h. Furthermore, the defined h_{1j} and h_{2j} satisfies: $h_{1j} = \text{argmin}_h\{ct_j(h) \in (k_1, k_2)\}$ and $h_{2j} = \text{argmax}_h\{ct_j(h) \in (k_1, k_2)\}$. Thus, $ct_j(h) \in (k_1, k_2)$ is equivalent to $h \in [h_{1j}, h_{2j}]$ and $P_h(h \in [h_{1j}, h_{2j}]) \geq 1-\alpha$. Therefore, $[h_{1j}, h_{2j}]$ is a $1-\alpha$ confidence interval for V_j.

A.1.6 Proof of Theorem 4. (1) First, we prove the convergence of $\mathcal{S}_j$. The main idea is to use the normal approximation of the Bernoulli distribution which is exact when $N \to \infty$, so that we can explicitly write k_1 and k_2 in terms of N and α. Then we take the limit $N \to \infty$ to conclude both k_1 and k_2 converge to $N/2$. Therefore, it follows that the average of samples between the k_1 and k_2 index converges to the median.

By applying the normal approximation of Bernoulli distribution, we have $\alpha/2 \approx \sum_{i=0}^{k_1} 0.5^N \binom{N}{i} \approx F_{\mathbb{N}(N/2,N/4)}(k_1)$ for large N, where $F_{\mathbb{N}(N/2,N/4)}$ is the CDF of normal distribution with mean $N/2$ and variance $N/4$. In addition, we can write $F_{\mathbb{N}(N/2,N/4)}(k_1) = F_{\mathbb{N}(0,N/4)}(k_1 - \frac{N}{2}) = F_{\mathbb{N}(0,1)}((k_1 - \frac{N}{2})/(\frac{\sqrt{N}}{2}))$. Therefore, $(k_1 - \frac{N}{2})/(\frac{\sqrt{N}}{2}) = -\mu_{\alpha/2}$, where $\mu_{\alpha/2}$ is the $\alpha/2$ upper quantile of normal distribution. By rearranging this equation, we have $k_1 \approx -\frac{\sqrt{N}}{2}\mu_{\alpha/2} + \frac{N}{2}$. Thus, $k_2 = N - k_1 \approx \frac{N}{2} + \frac{\sqrt{N}}{2}\mu_{\alpha/2}$. Therefore, k_1/N and k_2/N converges to $1/2$. Now, $\mathcal{S}_j = \sum_{i=k_1+1}^{k_2-1} e_{(i)j}/(k_2 - k_1 - 1) \geq \sum_{i=k_1+1}^{k_2-1} e_{(k_1)j}/(k_2 - k_1 - 1) = e_{(k_1)j} \to e_{(N/2)j} = V_j$ when $N \to \infty$. Similarly, $\mathcal{S}_j = \sum_{i=k_1+1}^{k_2-1} e_{(i)j}/(k_2 - k_1 - 1) \leq \sum_{i=k_1+1}^{k_2-1} e_{(k_2)j}/(k_2 - k_1 - 1) \to V_j$ when $N \to \infty$. Using the pinching theorem, we get $\mathcal{S}_j \to V_j$ when $N \to \infty$.

(2) Second, we compute the lower bound of the convergence rate for $\mathcal{S}_j$. The main idea is to upper bound the variance of $\mathcal{S}_j$, and then show the upper bound decreases in the speed of $O(1/N)$.

We will apply the *approximate summaries* theorem from Page 120 of Baglivo [6], which states that for any X_i sampled from some distribution, $\text{Var}(e_{(k)j}) \approx \frac{p(1-p)}{(N+2)(f_e(\theta))^2}$ where $p = \frac{k}{N+1}$, θ is the pth lower quantile and f_e is the PDF of e. Applying this theorem directly to compute the variance of $e_{(k_1+i)j}$ for i from 1 to $k_2 - k_1 - 1$, we get $\text{Var}(e_{(k_1+i)j}) \approx \frac{(k_1+i)(N-k_1+1-i)}{(N+1)^2(N+2)} \times \frac{1}{f_e(\theta)} \approx \frac{(\frac{N}{2} - \frac{\sqrt{N}}{2}\mu_{\alpha/2}+i)(\frac{N}{2} + \frac{\sqrt{N}}{2}\mu_{\alpha/2}-i)}{(N+1)^2(N+2)} \times \frac{1}{f_e(\theta)}$. Since i is in the order of $O(\sqrt{N})$, we have $\text{Var}(e_{(k_1+i)j}) \approx \frac{1}{4N} \times \frac{1}{f_e(\theta)}$. In addition, $p = \frac{k_1+i}{N+1} \to \frac{1}{2}$, which means $\theta = V_j$. Therefore, under the mild assumption that $f_e(V_j) > 0$, $\frac{1}{f_e(\theta)}$ converges to a constant. Combining this fact with the previous formula, we have $Var(e_{(k_1+i)j}) = O(1/N)$. Since $\text{Var}(A + B) = \text{Var}(A) + \text{Var}(B) + 2\,\text{Cor}(A, B) \leq (\sqrt{\text{Var}(A)} + \sqrt{\text{Var}(A)})^2$ for any A and B, using induction, we get $\text{Var}(\sum_{i=1}^{k} X_i) \leq (\sum_{i=1}^{k} \sqrt{\text{Var}(X_i)})^2$. By applying this formula on $\mathcal{S}_j$, we get $\text{Var}(\mathcal{S}_j) \approx \frac{1}{\mu_{\alpha/2}^2 N} \text{Var}(\sum_{i=k_1+1}^{k_2-1} e_{(i)j}) \leq \left(O(\frac{1}{\sqrt{N}}) \times O(\sqrt{N})\right)^2 \times O(\frac{1}{N}) = O(\frac{1}{N})$.

A.2 Additional Experimental Results

A.2.1 Examples. (1) An example to show the advantage of MeTFA compared to LEMNA. We provide an example predicted by the model as toxic as follows. The red words are the top 3 important words found by the attribution map, and the yellow words are the top 4-7 important words. As we can see, this text is misclassified by the model, and MeTFA tells that 'Dick' misleads the model while the vanilla LEMNA fails to find this.

LEMNA: "I'm pretty sure this is a joke. The other books for sale are How to raise a Jewish dog and Yiddish for Dick and Jane. It is an expensive and elaborate hoax. And, after the novelty wears off, not even that funny."

MeTFA-smoothed LEMNA:"I'm pretty sure this is a joke. The other books for sale are How to raise a Jewish dog and Yiddish for Dick and Jane. It is an expensive and elaborate hoax. And, after the novelty wears off, not even that funny."

(2) An example to show how the insertion and deletion calculate. The example is shown in 10. For insertion, the chow chow's predicted score increases while inserting the top $n\%$ important pixels highlighted by the attribution map, where n from 0 to 100. A more faithful attribution map highlights the supportive features more accurately, thus getting a higher area under the curve. On the opposite, we gradually delete the important pixels, and hence the predicted

Table 14: The mstd value of the MeTFA-smoothed Gradient and the vanilla Gradient for Densenet169.

$\mathbb{P}\backslash\mathbb{O}_n$	Normal	Uniform	Brightness	Avg
Normal	**0.0184**	**0.0164**	0.0155	**0.0167**
Uniform	0.0217	0.0185	0.0155	0.0186
Brightness	0.0457	0.0445	**0.0095**	0.0332
Vanilla Gradient	0.0495	0.0489	0.0140	0.0374

Table 15: The results with the robust faithfulness metric with $\mathbb{O}_n = N(0, 0.1)$. The tuple in the table is structured as (the score of MeTFA-smoothed explanation, the score of the vanilla explanation).

method	RI	RD	RO
ScoreCAM	(**0.7718**,0.6187)	(**0.2024**,0.2068)	(**0.5694**,0.4119)
RISE	(**0.6600**,0.4738)	(0.2645,**0.1620**)	(**0.3955**,0.3118)
IGOS	(**0.4295**,0.3288)	(0.0953,**0.0932**)	(**0.3342**,0.2356)

Table 16: The results of $std_{MeTFA-Grad}/std_{SG-Grad}$ for Densenet169 when $\mathbb{P}$ and $\mathbb{O}_n$ are both Uniform or Normal.

Sampling number	Uniform	Normal
10	0.9550	0.9473
30	0.9280	0.9175
50	0.9171	0.9052
70	0.9093	0.8985

Table 17: The results of $mstd_{MeTFA-Grad}/mstd_{SG-Grad}$ for VGG16 when $\mathbb{P}$ is Uniform and Normal.

N	Uniform	Normal
10	0.9361	0.9237
30	0.9374	0.9141
50	0.9478	0.9241
70	0.9603	0.9345

score of chow chow is reducing. A more faithful attribution map should have a lower area under the deletion curve. Thus, we can use the area under the insertion curve or deletion curve to reflect the accuracy of the attribution map to highlight the supportive features.

Table 10: The mstd value of the MeTFA-smoothed Gradient and the vanilla Gradient for Resnet50.

$\mathbb{P}\backslash\mathbb{N}$	Normal	Uniform	Brightness	Avg
Normal	**0.0176**	**0.0154**	0.0142	**0.0157**
Uniform	0.0208	0.0171	0.0142	0.0173
Brightness	0.0409	0.0403	**0.0104**	0.0305
Vanilla Gradient	0.0457	0.0445	0.0170	0.0357

Table 11: The mstd value of the MeTFA-smoothed LIME and the vanilla LIME for Resnet50.

$\mathbb{P}\backslash\mathbb{O}_n$	Normal	Uniform	Brightness	Avg
Normal	**0.0868**	**0.0752**	0.074	**0.0786**
Uniform	0.0907	0.0761	0.072	0.0796
Brightness	0.1524	0.1486	**0.0698**	0.1236
Vanilla	0.2148	0.1979	0.1193	0.1773

Table 12: The mstd value of the MeTFA-smoothed IGOS and the vanilla IGOS for Resnet50.

$\mathbb{P}\backslash\mathbb{O}_n$	Normal	Uniform	Brightness	Avg
Normal	**0.0278**	**0.0202**	0.0156	**0.0212**
Uniform	0.0308	0.0214	0.0131	0.02176
Brightness	0.0341	0.0258	**0.0093**	0.0231
Vanilla	0.0437	0.0335	0.0184	0.0318

Table 13: The mstd value of the MeTFA-smoothed ScoreCAM and the vanilla ScoreCAM for Resnet50.

$\mathbb{P}\backslash\mathbb{O}_n$	Normal	Uniform	Brightness	Avg
Normal	**0.0469**	**0.033**	0.0201	**0.033**
Uniform	0.0522	0.0344	0.0168	0.0341
Brightness	0.0583	0.0458	0.012	0.0387
Vanilla	0.0607	0.0465	**0.0112**	0.0395

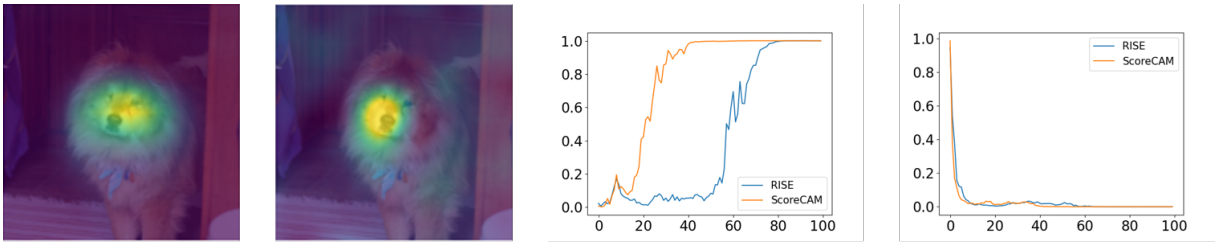

Figure 10: An example of the insertion metric and the deletion metric.

Table 18: The results of $std_{MeTFA-Grad}/std_{Grad}$ for Densenet169 under two settings: $\mathbb{P}$ and $\mathbb{O}_n$ are both Uniform or Normal.

α	Uniform	Normal
0.05	0.2558	0.2441
0.01	0.2553	0.2436
0.005	0.2552	0.2436
0.001	0.2552	0.2437
0.0005	0.2554	0.2439
0.0001	0.2557	0.2442

Table 19: The results for Toxic Comment dataset. The mstd values for LEMNA(500), MeTFA-smoothed LEMNA(500), smoothed LENMNA(500) when $\mathbb{P}$ and $\mathbb{O}_n$ are both P(0.5).

noise	mstd of LEMNA	mstd of MeTFA-smoothed LEMNA	mstd of smoothed LEMNA
P(0.5)	0.1531	**0.05488**	0.0667

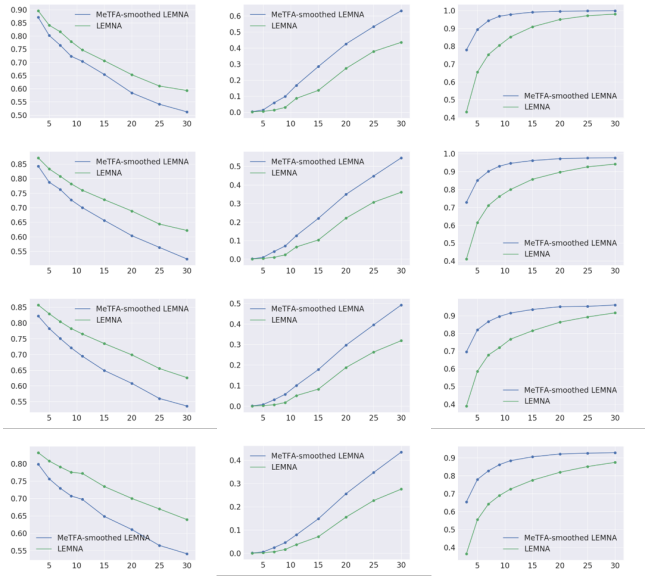

Figure 14: The results of IMDb Reviews dataset where the number of samples for LEMNA is 2000. The rows are the results of RFDT, RFAT and RST calculated with different $\mathbb{O}_n$. From top to bottom, $\mathbb{O}_n$ is P(0), P(0.3), P(0.5) and P(0.7), respectively. From left to right, the results are RFDT, RFAT and RST, respectively.

Algorithm 1: Generate MeTFA maps.

Parameters : number of sampled explanation for MeTFA n, vanilla explanation E, sampled distribution $\mathbb{P}$, target model M, target class c, target data I, confidence level α

Output: MeTFA-significant map $m1$, MeTFA-smoothed map $m2$, higher bound map $m3$ and lower bound map $m4$

/* line 1 to 7 sample explanations around I */

expl_list=[]

for $i \leftarrow 0$ **to** n **do**

  $inner\_noise \sim \mathbb{P}$

  $I_s = I + inner\_noise$

  $expl = E(M, I_s, c)$

  expl_list.append(expl)

end

/* line 8 to 15 generate $m1$ */

perform 1D clustering with all values in expl_list and get break value h

Compute $ct_j(h)$ for every feature j

$k_1 = \arg\max_k\{p\text{-value of } (V_j \le h) \le \alpha\}$

$k_2 = \arg\max_k\{p\text{-value of } (V_j \ge h) \le \alpha\}$

if $ct_j(h) \le k_1$, $m1_j = -1$

if $ct_j(h) \ge k_2$, $m1_j = 1$

else, $m1_j = 0$

/* line 15 to 19 generate $m2$, $m3$ and $m4$ */

k_1, k_2 are calculated as above

for every feature j, sort the value of expl_list

$m2_j$ is the average of the values between the k_1 th value and k_2 th value of expl_list$_j$

$m3_j$ is the k_2 th value of expl_list$_j$

$m2_j$ is the k_1 th value of expl_list$_j$

Return $m1$, $m2$, $m3$, $m4$

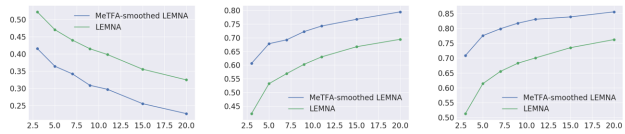

Figure 11: The results of of RFDT, RFAT and RST with different n where $\mathbb{O}_n$ is P(0.3). From left to right, there are the results of RFDT, RFAT and RST, respectively.

Algorithm 2: The RI calculation for MeTFA-smoothed explanation.

Parameters : n is the number of sampled explanation for MeTFA, k is the number of insertion for RI, the vanilla explanation E, the outer noise distribution $\mathbb{O}_n$, the inner noise distribution $\mathbb{P}$, the test dataset **D**, the target model M, the target class c

Output: RI

for *image I in* **D** **do**
  /* line 2 to 11 generate the MeTFA-smoothed explanation */
  expl_list=[]
  for $i \leftarrow 0$ **to** n **do**
    $inner\_noise \sim \mathbb{P}$
    $I_s = I + inner\_noise$
    $expl = E(M, I_s, c)$
    expl_list.append(expl)
  end
  k_1, k_2 are calculated from Theorem 3
  for every pixel, sort the value of expl_list
  for every pixel, take the average of the values between the k_1 th value and k_2 th value and get MS_expl
  /* line 12 to 17 calculate the RI metric */
  for $i \leftarrow 0$ **to** k **do**
    $outer\_noise \sim \mathbf{O}_n$
    $I_n = I + outer\_noise$
    insertion = Insertion(MS_expl, I_n, M, c)
  end
  /* use the average of k insertion as the approximation of the expectation */
  take the average of k insertion values to get RI value
end
Return the average of RI values

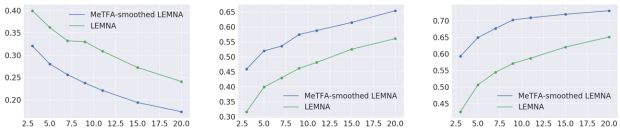

Figure 12: The results of of RFDT, RFAT and RST with different n where $\mathbb{O}_n$ is $\mathrm{P}(0.7)$. From left to right, there are the results of RFDT, RFAT and RST, respectively.

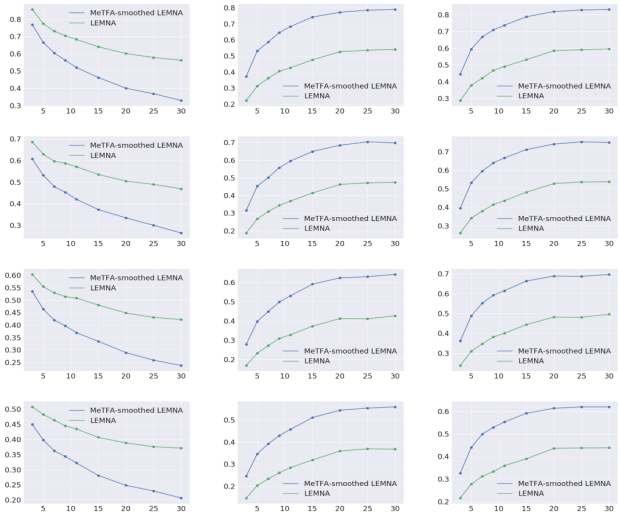

Figure 13: The results of Toxic Comment dataset where the number of samples for LEMNA is 500. The rows are the results of RFDT, RFAT and RST calculated with different $\mathbb{O}_n$. From top to bottom, $\mathbb{O}_n$ is $\mathrm{P}(0)$, $\mathrm{P}(0.3)$, $\mathrm{P}(0.5)$ and $\mathrm{P}(0.7)$, respectively. From left to right, the results are RFDT, RFAT and RST, respectively.