1-1-2023

Adversarial Explanations for Text-Based Local Surrogate Models

Christopher Burger
University of Mississippi

ADVERSARIAL EXPLANATIONS FOR TEXT-BASED LOCAL SURROGATE MODELS

MASTER'S THESIS

A Thesis
presented in partial fulfillment of requirements
for the degree of Master of Science
in the Department of Computer & Information Science
The University of Mississippi

by

Christopher Burger

August 2023

ABSTRACT

LIME has emerged as one of the most commonly referenced tools in explainable AI (XAI) frameworks that is integrated into critical machine learning applications–e.g., healthcare and finance. However, its stability remains little explored, especially in the context of text data, due to the unique challenges when it comes to perturbing the underlying data, data that requires a stricter measure of semantic and syntactic consistency. To address these challenges, we first evaluate the inherent instability of LIME on text data to establish a baseline, and then propose a novel algorithm XAIFOOLER to perturb text inputs and manipulate explanations that casts investigation on the stability of LIME as a text perturbation optimization problem. XAIFOOLER conforms to the constraints to preserve text semantics and original model prediction with an enforcement of small, similar perturbations, and introduces Rank-biased Overlap (RBO) as a key part to guide the optimization of XAIFOOLER which is an alternative to more commonly used similarity measures that satisfies all the requirements necessary for an idea explanation similarity measure. Extensive experiments on real-world text datasets demonstrate that XAIFOOLER significantly outperforms all baselines by large margins in its ability to manipulate LIME's explanations with high semantic preservability.

ACKNOWLEDGEMENTS

My sincerest thanks to my advisor, Thai Le, and our collaborator Lingwei Chen for their dedicated effort in the creation and refinement of this thesis work into our paper "**Are Your Explanations Reliable?" Investigating the Stability of LIME in Explaining Text Classifiers by Marrying XAI and Adversarial Attack**.

Additionally, I thank my committee members Timothy Holston and Charlie Walter for generously donating their time to serve on the Master's committee.

TABLE OF CONTENTS

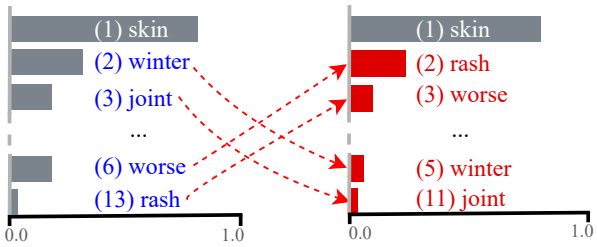
1 INTRODUCTION

1.1 Explainability and Stability in Machine Learning

Machine learning has witnessed extensive research, leading to its continually increasing capability in predicting a wide variety of phenomena [23]. Unfortunately, this increased effectiveness has come at the cost of comprehending the inner workings of the resulting models. To address this challenge, explainable AI (XAI), also known as interpretable AI [30], has emerged as a discipline focused on understanding *why* a model makes the predictions it does. XAI continues to grow in importance due to both legal and societal demands for elucidating the factors contributing to specific model predictions.

While explainability in AI as a general concept has yet to fully coalesce to an accepted set of definitions [9], the concept of *Stability* occurs throughout the discussions and continues to remain prominent [20, 5, 33]. A stable explanation refers to one where small changes to the input should have a corresponding small effect on the output. In other words, similar inputs to a model should produce similar explanations. Lack of stability in an explanatory method undermines its trustworthiness [9], and renders all subsequent explanations suspect. This lack of trustworthiness is one reason why the adoption of AI in disciplines like healthcare has progressed slower than in other disciplines [19].

Previous work on stability in XAI has largely focused on models where function-based continuity criteria naturally apply to tabular and image-based data [1, 33, 9]. However, text data in natural language is not so amenable to such direct quantification. The process of generating appropriate perturbations to test stability in text explanations remains little explored [11] as unlike perturbations in tabular or image-based data, text perturbations pose unique challenges to satisfy



"I have a skin rash that ~~***gets***~~ ***becomes*** *worse in the winter. I have to moisturize more* ~~***regularly***~~ ***consistently*** *and have [...]. I also have joint pain."*

Figure 1.1: With only two perturbations, XAIFOOLER significantly demoted "joint pain" symptom from top-3 to 11th-rank (left→right figure) while maintaining both the original prediction label and the clarity and meaning of the original text.

necessary constraints of semantic similarity and syntactic consistency. Violation of these constraints can create a fundamentally different meaning conveyed by the document. Small changes can therefore have large consequences, unlike in something like image data where the change of a few pixels can often be effectively indistinguishable to humans. Under the assumption the explanatory algorithm works correctly, these perturbations may alter the resulting explanation, where quantifying the difference between explanations also needs to characterize their properties, such as feature ordering and weights within explanations as presented in Fig. 1.1.

We explore the stability of explanations generated on text data via LIME [25], which is a widely used explanatory algorithm in XAI frameworks. We first examine the inherent stability of LIME by altering the number of samples generated to train the surrogate model. Using this baseline, we then propose XAIFOOLER, a novel algorithm that perturbs text inputs and manipulates explanations to investigate in depth the stability of LIME in explaining text classifiers. Given a document, XAIFOOLER proceeds with iterative word replacements conforming to the specified constraints, which are guided by Rank-biased Overlap (RBO) that satisfies all the desired characteristics for explanation similarity measure. As such, XAIFOOLER yields advantages that only small perturbations on the least important words are needed, while the semantics and original prediction of the input are preserved yet top-ranked explanation features are significantly shifted.

Fig. 1.1 shows an example that XAIFOOLER only performs two perturbations (i.e., "gets" → "becomes" and "regularly" → "consistently"), but effectively demotes the top-3 features–e.g., "joint" (3rd-rank)→(11th-rank), that explains the "joint pain" symptom without changing the prediction of *peptic ulcers* as the symptom's cause. Our contributions are summarized as follows.

- We assess the inherent instability of LIME as a preliminary step towards better understanding of its practical implications, which also serves as a baseline for subsequent stability analysis.
- We cast investigation on the stability of LIME as a text perturbation optimization problem, and introduce XAIFOOLER with thoughtful constraints and explanation similarity measure RBO to generate text perturbations that effectively manipulates explanations while maintaining the class prediction in a cost-efficient manner.
- We conduct extensive experiments on real-world text datasets, which validate that XAIFOOLER significantly outperforms all baselines by large margins in its ability to manipulate LIME's explanations with high semantic preservability.

2 BACKGROUND

2.1 Surrogate Explanation Models

Surrogates are a class of interpretable models that are used to explain the predictions of an arbitrary target black-box model. Global surrogate models train a model such as a Decision Tree as an explainable substitute for the target model [20]. Local surrogate models instead concentrate on explaining individual predictions, rather than attempting to maximize global fidelity. As the quality of the global approximation is generally limited (otherwise the global surrogate would be used in place of the target model), the trade-off for better local explanations is a reasonable one to make. In practice, local explanations for specific predictions of interest are also more immediately-important and intuitive to the end-users.

2.2 LIME

We adopt Local Interpretable Model-agnostic Explanations (LIME) [25] as our target explanatory algorithm. LIME has been a commonly researched and referenced tool in XAI frameworks, which is integrated into critical ML applications such as finance [10] and healthcare [14, 6]. To explain a prediction, LIME trains a shallow, explainable surrogate model such as Logistic Regression or a Decision Tree on training examples that are synthesized within the vicinity of an individual prediction. The resulting explanation is a subset of coefficients of this surrogate model that satisfies the fundamental requirement for interpretability. In NLP, explanations generated by LIME are features–e.g., words, returned from the original document, which can be easily understood even by non-specialists.

2.3 XAI Stability

Existing research work on XAI stability has a predominant emphasis on evaluating models using tabular or image data across various interpretation methods, which often use small perturbations to the input data to generate appreciable different explanations [1, 9, 1], or generate explanations that consist of arbitrary features [28].

LIME specifically has been analyzed for its efficacy. Garreau et al. first investigated the stability of LIME for tabular data [7, 8], which showed that important features can be omitted from the resulting explanations by changing parameters. They extended the analysis to text data later [18] but only on the fidelity instead of *stability* of surrogate models. Other relevant works in text domain include [11], which utilized gradient-based explanation methods, assuming white-box access to the target model and hence not realistic because model parameters are often inaccessible in practice; and [27], which revealed that LIME's explanations are unstable to black-box text perturbations. However, it adopts a very small sampling rate for LIME (n=500) which we later demonstrate to significantly overestimate LIME's instability. Moreover, its experiment settings are not ideal as it allows the perturbations of top-ranked predictive features, which naturally change the resulting explanations.

Although existing works have showed that LIME is sensitive to small perturbations in the target model's inputs or parameters, they do not answer the fundamental question of whether LIME itself is inherently unstable *without* any changes to the input or model's parameters. Answers to this *"what is the inherent instability of* LIME*"* question would establish a meaningful baseline that helps us better evaluate our stability analysis.

3 INHERENT INSTABILITY OF LIME

We begin by first assessing the inherent instability of LIME. Our aim is to determine if LIME produces inconsistent results even when no changes are made to the target model's inputs or parameters. Let d be a document whose prediction under a target model $f(\cdot)$ is to be explained. A simplified process of LIME's explanation generation for $f(d)$ is as follows.

- *Step 1:* Generate perturbed document d_i by randomly selecting k words from d and remove all of their occurrences from d.
- *Step 2:* Repeat *Step 1* to sample n different perturbations $\{d_i\}_{i=1}^{n}$.
- *Step 3:* Train an explainable model $g(\cdot)$ via supervised learning with features d_i and labels $f(d_i)$.

Here n is the *sampling rate* which determines the number of local training examples needed to train the surrogate model $g(\cdot)$. The sampling rate n greatly influences the performance of LIME. Intuitively, a small n often leads to insufficient amount of data for training a good $g(\cdot)$. In contrast, a large n may result in better $g(\cdot)$ but also adversely increases the runtime due to a large number of inference passes required to collect all $f(d_i)$. As the results of this process is effectively a classic sampling problem, $g(\cdot)$ eventually converges towards a fixed model as $n \to \infty$. And as such there are eventual thresholds of diminishing returns as n grows larger. For the new local model $g(\cdot)$, the estimated local classifier weights for the features d_i are generated from the probability of the perturbed document with respect to the true output and the proportion of variation of the perturbed document away from the original. This is calculated as $1 - \frac{Number\ of\ Words\ Removed}{Total\ Words}$.

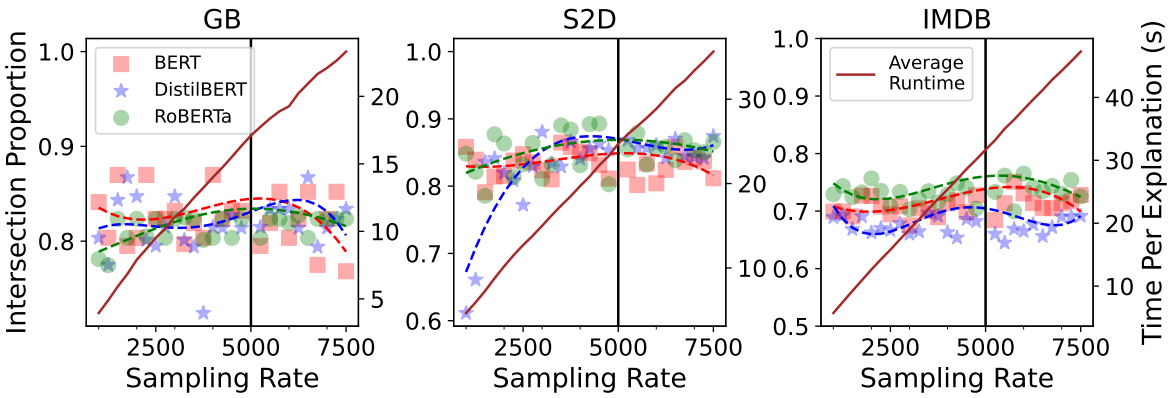


Figure 3.1: Inherent explanation instabilities for each model and dataset.

Dataset	Mean	Median	Min
GB	82.0% (↓ Δ18%)	82.0%	75.5%
S2D	84.0% (↓ Δ16%)	84.4%	72.9%
IMDB	71.5% (↓ Δ28.5%)	70.4%	67.3%

Table 3.1: Statistics of explanation similarities when using different alternate sampling rates compared against the base explanation with default sampling rate n=5000.

To test the inherent instability of LIME, we first select an arbitrary number of documents and generate explanations at different sampling rates n from 1000 to 7500 (steps of 250) for three state-of-the-art classifiers, including BERT [3], RoBERTa [16], DistilBERT [26], trained on three corpus varying in lengths. Then, we compare them against the base explanation generated with the default sampling rate (n=5000). Table 3.1 shows that altering the sampling rate from its default value results in significant dissimilarities in the explanations (↓16%–↓28.5% in similarity on average), which observe to be more significant on longer documents. This happens because longer documents generally require more training examples to refine the local model $g(\cdot)$ to better approximate $f(\cdot)$. Fig. 3.1 gives us a closer look into such instability, which further shows evidence that in all datasets, both bounded variation and diminishing returns converge as the sampling rate increases. However, with small sampling rates n–e.g., $n < 4000$ in IMDB dataset, slight changes in n can produce significantly dissimilar explanations.

Overall, LIME itself is inherently unstable, especially when n is small, even without any modifications to the target model or its inputs. In contrast to small sampling rates–e.g., n=500, adopted in existing analytical works [27], our analysis implies that n should be sufficiently large according to each dataset to maximize the trade-off between computational cost and explanation stability.

This inherent instability is significant in practice due to the fact that many important AI systems–e.g., ML model debugging [15] or human-in-the-loop system [22], use surrogate methods such as LIME as a core component. This instability also has implications on software security where malicious actors can exploit this instability and force AI models to produce unfaithful explanations. To push this instability test to its limit and to thoroughly comprehend the effects of LIME's instability in practice, we investigate the robustness of LIME in depth by formulating this investigation as an adversarial text attack optimization problem.

4 PROBLEM FORMULATION

4.1 Motivation

The property of whether a model's decisions are reasonable is the primary goal of XAI. We as humans have an intuitive, but not necessarily articulable, understanding of the relationships that govern the decision to be predicted, even if only the understanding that certain features cannot possibly be relevant to the decision making process. We appeal to this intuition when determining if an explanation is reasonable. Unfortunately the threshold for what is reasonable is both task and consequence dependent, and so this concept becomes very difficult to rigorously define in general terms, especially in a quantitative sense. Fortunately for text models it is easy to satisfy this reasonableness. The features are subsets of the input text, often just single words. Local surrogate models like LIME satisfy this property well, the result is just an ordered list of words. Humans are naturally quite capable at textual analysis tasks, most individuals fluent in the language the model is trained with can perform a task like sentiment analysis with good accuracy. However we need to convert the idea of reasonable into something that can be processed computationally. In doing so we will avoid setting a specific threshold for a successful attack, as what defines success is subject to significant personal interpretation. Instead we will treat the process as an optimization problem and run the algorithm until certain constraints related to quality of the perturbed document no longer hold.

4.2 Creating the Objective Function

Our goal is to determine how much a malicious actor can manipulate explanations generated by LIME via text perturbations–i.e., minimally perturbing a document such that its explanation significantly changes. Specifically, a perturbed document d_p is generated from a base document d_b, such that for their respective explanations e_{d_p} and e_{d_b} we *minimize their explanation similarity*:

$$d_p = \underset{d_p}{\operatorname{argmin}} \ \boldsymbol{Sim}_{\mathbf{e}}(e_{d_b}, e_{d_p}), \tag{4.2.1}$$

where $\boldsymbol{Sim}_{\mathbf{e}}(\cdot)$ is the similarity function between two explanations. To optimize Eq. (4.2.1), our method involves a series of successive perturbations within the original document as often proposed in adversarial text literature. In a typical adversarial text setting, malicious actors aim to manipulate the target model's prediction on the perturbed document, which naturally leads to significant changes in the original explanation. But this is not meaningful for our attack; thus, we want to *preserve the original prediction* while altering only its explanation:

$$f(d_b) = f(d_p), \tag{4.2.2}$$

Trivially, changing words chosen arbitrarily and with equally arbitrary substitutions would, eventually, produce an explanation different from the original. However, this will also likely produce a perturbed document d_p whose semantic meaning drastically differs from d_b. Thus, we impose a constraint on the semantic similarity between d_b and d_p to ensure that the perturbed document d_p does not alter the fundamental meaning of d_b:

$$\boldsymbol{Sim}_{\mathbf{s}}(d_b, d_p) \geq \delta, \tag{4.2.3}$$

where $\boldsymbol{Sim}_{\mathbf{s}}(\cdot)$ is the semantic similarity between two documents and δ is a sufficiently large hyper-parameter threshold.

Even with the semantic constraint in Eq. (4.2.3), there can still be some shift regarding the context and semantics of d_b and d_p that might not be impeccable to humans yet cannot algorithmically captured by the computer either. Ideally, the malicious actor wants the perturbed document d_p to closely resemble its original d_b. However, as the number of perturbations grows larger, the document retains less of its original context and meaning. To address this issue, we impose a

maximum number of perturbations allowed through the use of a hyper-parameter ϵ as follows:

$$i \leq \epsilon * |f|, \tag{4.2.4}$$

where an accepted i-th perturbation will have replaced i total number of words (as each perturbation replaces a single word) and $|f|$ is the total number of *unigram bag-of-words* features in f.

We can now generate perturbations in a way that optimizes our ability to maintain the intrinsic meaning of the original document. If we are to manipulate the explanation (Eq. (4.2.1)) while maintaining both Eq. (4.2.2), Eq. (4.2.3) and Eq. (4.2.4), it is trivial to just replace a few most important features of f. However, in practice, changing the most important features will likely result in a violation to constraint in Eq. (4.2.2). Moreover, this will not provide any meaningful insights to analysis on stability in that we want to measure how many changes in the perturbed explanation that correspond to small (and not large) alterations to the document. Thus, the set of top k feature(s) belonging to the base explanation e_{d_b} must appear in the perturbed explanation e_{d_p}:

$$e_{d_p} \cap c \neq \varnothing \quad \forall\, c \in e_{d_b}[:k]. \tag{4.2.5}$$

Overall, our objective function is as follows.

OBJECTIVE FUNCTION: Given a document d_b, a target model f and hyper-parameter δ, ϵ, k, our goal is to find a perturbed document d_p by optimizing the objective function:

$$\begin{aligned} d_p = \underset{d_p}{\operatorname{argmin}} \;\; & \boldsymbol{Sim}_{\mathbf{e}}(e_{d_b}, e_{d_p}), \\ \text{s.t.} \;\; & f(d_b) = f(d_p), \\ & \boldsymbol{Sim}_{\mathbf{s}}(d_b, d_p) \geq \delta, \\ & i \leq \epsilon * |f|, \\ & e_{d_p} \cap c \neq \varnothing \quad \forall\, c \in e_{d_b}[:k] \end{aligned} \tag{4.2.6}$$

5 DETERMINING EXPLANATION SIMILARITY

5.1 Similarity Measures for Explanations

The core of the objective function (4.2.6) is the similarity measure $\boldsymbol{Sim}(\cdot)$. Without a carefully chosen measure no useful results can be collected. Too coarse and no change in similarity becomes apparent, too fine and minute changes result exaggerated differences. To begin we define a series of properties that allow us a balance in sensitivity of the measure while being suitable for the unmodified output of the explanatory algorithm. The default output of LIME is a ranked list of unique features, features that are subsets of the document to be explained. These features are ordered from most to least important away from the true output of the original model.

Define $\boldsymbol{Sim}_{\mathbf{e}}$ as our explanation similarity function used to compared two explanations e_{d_b} and e_{d_p}. Then $\boldsymbol{Sim}_{\mathbf{e}}$ should satisfy the following properties.

(A) Positional Importance [13]. We require that a relative ordering be imposed on the features–i.e., higher-ranked features should be accorded more influence within $\boldsymbol{Sim}_{\mathbf{e}}(\cdot)$. That is, moving the 1st-ranked feature to rank 10 should be considered more significant than moving the 10th-ranked feature to rank 20. Moreover, this requirement also accounts the fact that end-users often consider only the top k most important *and not all of the* features [29] in practice, and thus $\boldsymbol{Sim}_{\mathbf{e}}(\cdot)$ should be able to distinguish the ranking of different features.

(B) Weighted Features. This is a strengthening of the positional importance by considering not only discrete positional rankings such as 1st, 2nd, 3rd, but also their continuous weights such as the

feature importance scores provided by LIME. This provides us more granularity in the resulting explanation similarities and also better signals to guide our optimization process for Eq. (4.2.6).

(C) Disjoint Features. We require that disjoint lists can be compared. Consider a single word replacement of word w with word w' in a document d_b with base explanation e_{d_b} and perturbed explanation e_{d_p} generated from $d-w+w'$. The perturbed explanation e_{d_p} cannot contain the word w or any subset of d_b containing w as a possible feature.

(D) Unequal List Lengths. Similarly to (C) but if w' already exists in the original explanation e_b, the length of e_p will be different from e_b due to d_p having one less unique unigram feature.

5.2 Existing Similarity Measures

Given our desired properties, we now proceed to search for an ideal explanation similarity measure for $\boldsymbol{Sim}_{\mathbf{e}}$ from some of commonly used measures on ranked lists. We can divide these into two groups, *weight-based* which are distance measures focused entirely on the weight associated with the feature, and *feature-based* which concern themselves with the feature itself and often the positional importance of the feature.

Weight-based Measures

$\mathbf{L_p}$ Distance

The familiar L_p distances discard the feature order entirely and instead concentrate on their associated weights. In other words, L_p fails to consider positional importance and instead concentrates on the weights associated with the features. While certain formulations exist that allow weighting the elements (of which the elements themselves are the weights of the local classifier), the L_p distances still lack the capacity to handle differing ordered-list lengths. For example, we can encounter a scenario where the order of the features is identical, but the weights have shifted enough to provide a substantial L_p distance. Consider a list of features $F_1 = [a, b, c]$ and $F_2 = [a, b, c]$ with weights $W_1 = [3, 2, 1]$ and $W_2 = [4, 3, 2]$. Clearly the L_p distance here will be non-zero, but the

position of the features remains identical. Does the change in total weight matter? It may, and L_p is well specified if that is the case. However, we may not necessarily understand just how important the differences between weights are. Because of this, we do not wish to set a threshold for significance between the two explanations purely on a distance between the weights. We instead appeal to the position of the features, a coarse form of weighting, and use our own scheme for weight importance that concentrates the weights in an easily tuned manner of our own choosing. Despite not satisfying our criteria, $\mathbf{L_p}$ is common enough that we will use it as a comparative metric.

Location of Mass (LOM) Measure

The LOM measure can be considered an improvement upon the L_p distance as it not only determines that the weight has shifted but provides insight into where it is accumulating. However, a change in LOM does not imply an actual change in the relative order of features. If the location (often the center / median) of the cumulative weights has shifted, then more weight has been assigned to or away from certain features and so the explanation must then be different. LOM has issues similar to L_p in that the actual location of mass may not shift despite weight being changed significantly among many features, often brought about due to a large proportion of the sum total explanatory weight being associated with very few features. For example, consider the lists of features $F_1 = [a, b, c, d, e]$ and $F_2 = [a, b, c, d, e]$ with weights $W_1 = [1, 0, 5, 0, 1]$ and $W_2 = [1.3, 1.2, 2, 1.2, 1.3]$ Both collections of features possess the same total weight, and the same location of mass (center). But F_2 has a significantly different assignment of weights and it is reasonable to conclude that this explanation is different.

Additionally, for certain explanations that are highly concentrated in weight it may not be feasible to find perturbations that are able to distribute this among other features to the extent necessary to move the location of mass. For example, let $W_3 = [1, 2, 1, 1000, 2, 2, 1]$. We would have to redistribute the overwhelming majority of the weight to the other features in order to move this location of mass away from index 3.

Feature-based Measures

Jaccard Index

The Jaccard Index compares two sets by computing the ratio of the shared elements between the sets to the union of both sets resulting in a intuitive measure of similarity. Being a strictly set-based measure, the Jaccard index lacks positional importance and feature weighting (extensions exist to allow weight), making it unsuitable for our problem. More specifically, the Jaccard Index preserves no order between two lists of feature explanations. For collections of features $F_1 = [a, b, c]$ and $F_2 = [c, a, b]$ the Jaccard Index of F_1 and F_2 J(F_1,F_2) = 1 despite no pair of values having an equal position. These lists are clearly different, but the Jaccard Index does not have the capacity to differentiate between them. Additionally, the Jaccard Index assigns an equal value, related to the total number of elements, for any dissimilarity between the lists. For example, let

$$F_1 = [a, b, c, ..., x, y, z]$$
$$F_2 = [\alpha, b, c, ..., x, y, z]$$
$$F_3 = [a, b, c, ..., x, y, \omega]$$

With F_1 being ordered by weight in decreasing importance.

Finally let weight(α) < weight(a) and weight(y) > weight(ω) > weight(z).

Then we have J(F_1, F_2) = J(F_1, F_3).

Clearly the similarity between F_1 and F_2 should be considered less than F_1 and F_3 as we have additional information pertaining to each feature's importance.

General Correlation Measures

There are multiple variants of the general correlation measure, we consider the two of the most popular, Kendall's τ & Spearman's ρ. They are commonly used for determining the similarity

of two ranked lists and the central idea of both is the order of the features. Whereas τ uses the number of pairwise disagreements between the two lists of features, ρ is based on the difference between the two ranks of each observation. As both specific cases the generalized correlation coefficient they are subject to very similar strengths and weaknesses. The weaknesses being that they disregard any ranking weights, and lack the capacity to handle unequal list lengths and disjoint features. Remedies are also available–e.g., [13], but they do not fulfill all the requirements and can result in information loss.

More specifically, Kendall's τ & Spearman's ρ are subject to similar issues affecting the Jaccard Index. Both τ and ρ can handle the first scenario, while the Jaccard Index could not. That is, for $F_1 = [a, b, c]$ and $F_2 = [c, b, a]$ then Kendall(F_1,F_2) $\neq 1$ and Spearman(F_1,F_2) $\neq 1$. However, τ and ρ are similar to Jaccard in that the difference between weighted features has not been taken into account. For example, let

$$F_1 = [a, b, c, ..., x, y, z]$$
$$F_2 = [b, a, c, ..., x, y, z]$$
$$F_3 = [a, b, c, ..., x, z, y]$$

Then Kendall(F_1, F_2) = Kendall(F_1, F_3), and Spearman(F_1, F_2) = Spearman(F_1, F_3).

The exchange between features a and b should be considered more important than the exchange between features y and z. The resulting value is also related to the size of the lists, and a single exchange of two adjacent becomes less important as the size of the lists grows larger.

Rank-biased Overlap

So far, none of the mentioned measures satisfies all requirements. Instead we appeal to *Rank-biased Overlap* (RBO) [31] which well fits our criteria. RBO is a feature-based comparison measure that combines the benefits of the set-based approaches while retaining the positional information and capacity for feature weightings. The weights assigned to the features are controlled by a convergent series where the proportion of weights associated with the first k features is determined

Feature / Measure	Positional Importance	Feature Weighting	Disjoint Features	Unequal-Length List
Jaccard Index		✓*	✓	✓
Kendall's τ	✓	✓*		
Spearman's ρ	✓	✓*		
L_p		✓	✓	
Center of Mass		✓	✓	
†RBO	✓	✓	✓	✓

†RBO: Rank-biased Overlap; *: customized formulas exist

Table 5.1: Comparisons among explanation similarity measures with RBO satisfying all the requirements.

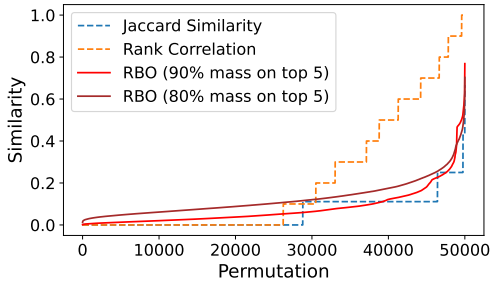


Figure 5.1: Simulation on 50K permutations of 50 features: RBO outputs smoothed, richer signals while other measures return zero similarity for over half the permutations and show poorer delineation among top-k features, resulting in the non-smoothed, step-wise behavior.

by a hyper-parameter $p \in (0, 1)$. As $p \to 0$ more weight is assigned to the topmost features, while as $p \to 1$ the weight becomes distributed more evenly across all possible features.

Fig. 5.1 illustrates the comparison between RBO, Jaccard, and Kendall/Spearman measures, which highlights two advantages of RBO. First, RBO allows features outside the top k some small influence on the resulting similarity. Second, the weighting scheme associated with RBO allows more granularity when determining similarity by applying weight to the top k features (Fig. 5.1), which is lacking with the other measures. In other words, RBO provides richer signals to guide the greedy optimization step of XAIFOOLER. Moreover, RBO allows us to decide how much distribution mass to assign to the top-k features (Fig. 5.1 with 90% & 80% mass and Fig. 5.2 with top-k parameters of 2, 3, and 5). This enables us to focus on manipulating the top-k features that the end-users mostly care about while not totally ignoring the remaining features.

Our use of RBO follows from our choice of desired attributes in Sec. 5.1. Here we provide examples to justify our selection of similarity measure expanding on the summary in Table 5.1.

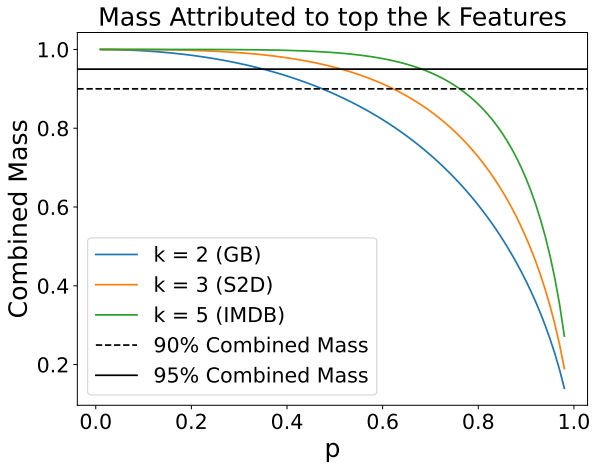


Figure 5.2: Mass associated with the top k features for different values of RBO's hyper-parameter p.

Note that the weight-based distances L_p, and LOM are different enough from the structure of the feature-based measures like RBO that direct comparison is difficult. Their construction gives them particular strengths but they fundamentally are unable to satisfy our desired attributes. They remain included as important metrics used for comparison against the other similarity measures.

However, RBO can be compared directly to the Jaccard and Kendall / Spearman similarities. We do so in Fig. 5.1 where we demonstrate two of RBO's advantages. First, RBO provides some capacity for determining similarity when using features not within the selected top k. Second, the weighting scheme associated with RBO allows more granularity when determining similarity; in other words, the similarity output is smoother due to the weight assignment, which is lacking with the other measures. Here a fixed list of 50 features is generated, which is then uniformly shuffled 50,000 times. The results are plotted according to the similarity output from each of the measures with respect to the top 5 features. We see RBO allows some small amount of similarity to remain desipte assigning the significant majority of the mass to the top 5 features. The other measures return 0 similarity for over half the permutations, as our requirement for a concise explanation allows us only to concentrate on the top few features. The other measures show poorer delineation between important features (as determined by position in the top 5), resulting in the step-wise behavior.

6 ALGORITHM

6.1 XAIFOOLER

To solve the aforementioned objective function (Eq. (4.2.6)), we propose XAIFOOLER, a novel greedy-based algorithm that can effectively manipulate explanations of LIME via text perturbation.

Algorithm 1 Adversarial Explanation Generation

1: *Input:* **target model** f, **Original Document** d_o, **Base Explanation** e_b, **Maximum Perturbation Threshold** p_t, **Current Perturbations** p_c **Current Similarity** s
2: *Output:* **Perturbed Document** d_p, **Perturbed Explanation** e_p, **Updated Similarity** s
3: *Initialize:* $e_p \leftarrow e_b$, $i \leftarrow 0$, $s \leftarrow 1$, $p_c \leftarrow 0$
4: *Initialize:* I $\leftarrow$ indices of replacement candidates that satisfy C
5: **while** $I \neq \varnothing \; and \; p_c < p_t$ **do**
6: s_i = RBO(d_p, perturb($d_p[I[i]]$)
7: **if** $s_i < s$ **then**
8: $s \leftarrow s_i$
9: $e_p \leftarrow$perturb($d_p[I[i]]$)
10: **end if**
11: $I[i] = \varnothing$
12: $i \leftarrow i + 1$
13: **end while**
14: **return** (d_p, e_p, s)

XAIFOOLER Algorithm

Algorithm 1 describes XAIFOOLER in two steps. First, given an input document, it decides which words to perturb and in what order (Ln. 4). Next, it greedily replaces each of the selected word with a candidate that (i) best minimizes the explanation similarity via $\boldsymbol{Sim}_{\mathbf{e}}$ (where our similarity measure is RBO) and (ii) satisfies all the constraints in Sec. 4.2 until reaching the maximum number of perturbations (Eq. (4.2.4)) (Ln. 5–13).

Step 1: Greedy Word Selection and Ordering. We disregard all stopwords and the top-k important features based on their absolute feature importance scores returned from LIME. We then order the remaining words in the original document d_b in descending order according to how many changes they make in the original prediction when they are individually removed from d_b. Intuitively, we prioritize altering features of lower predictive importance to signify the instability of LIME (Eq. 4.2.5).

Step 2: Greedy Search with Constraints. We subsequently replace each word in the list returned from Step 1 with a list of candidates and only keep those that help decrease the explanation similarity $\boldsymbol{Sim}_{\mathbf{e}}$ (Alg. 1, Ln. 6–10). To satisfy Eq. (4.2.2, 4.2.3, 4.2.4, 4.2.5), we only accept a perturbation if it satisfies these constraints, and at the same time improves the current best explanation similarity. To reduce the search space of replacements, we only select replacement candidates that maintain the same part-of-speech function and also within a similarity threshold with the word to be replaced in counter-fitted Paragram-SL999 word-embedding space [32] as similarly done in prior adversarial text literature (TextFooler [12]). As a sanity check, we also constrain the final perturbed document to result in at least one of the top k features decreasing in rank.

7 EXPERIMENTAL RESULTS

7.1 Preliminaries

Datasets and Target Models.

We experiment with three datasets: sentiment analysis (IMDB) [17], symptom to diagnosis classification (S2D) (from Kaggle) and gender bias classification (GB) [4], of varying averaged lengths (230, 29, 11 tokens), and number of labels (2, 21, 2 labels) (see Table 7.1). Each dataset is split into 80% training and 20% test sets. We use the training set to train three target models, namely DistilBERT [26], BERT [3] and RoBERTA [16], achieving around 90%–97% in test prediction F1 score.

Baselines.

We compare XAIFOOLER against four baselines, namely (1) *Inherency*: the inherent instability of LIME due to random synthesis of training examples to train the local surrogate model (Sec. 3); (2) *Random*: randomly selects a word to perturb and also randomly selects its replacement from a list of nearest neighbors (3) *Location of Mass (LOM)*: inspired by [27], similar to XAIFOOLER but uses COM (Sec. 5.2) as the explanation similarity function; (4) L_p: similar to XAIFOOLER but uses L_2 (Sec. 5.2) as the explanation similarity function.

Evaluation Metrics.

Similar to prior works, we report the explanation changes of top-k features using: *Absolute Change* in ranking orders (*ABS*↑); *Rank Correlation* (*RC*↑), which is calculated by the formula:

Dataset	# Tokens	# of Labels	DistilBERT	BERT	RoBERTa
IMDB	230	2	0.91	0.92	0.94
S2D	29	21	0.94	0.93	0.97
GB	11	2	0.90	0.90	0.89

Table 7.1: Dataset statistics and prediction performance (F1 Score) of target classification models on test set.

$1-max(0,$ Spearman-Correlation$(e_{d_b}[:k], e_{d_p}[:k])$; Intersection Ratio (*INS*↓)–i.e., ratio of features remained in top k after perturbation. Moreover, we also report the semantic similarity between d_b and d_p (*SIM*↑) by calculating the cosine-similarity of their vectors embedded using USE [2]; and the naturalness of d_p by calculating its perplexity score (*PPL*↓) using large language model GPT2-Large [24] as a proxy. Arrow ↑ and ↓ denote *the higher, the better* and *the lower, the better*, respectively.

7.2 Implementation Details

Parameter Selection

We use Fig. 5.2 to select the hyper-parameter p values of RBO that correspond to 90%, 95%, 95% mass concentrating in the top-5 ($k \leftarrow 5$), top-3 ($k \leftarrow 3$) and top-2 ($k \leftarrow 2$) features for IMDB, S2D and GB dataset, respectively. Based on Sec. 3, we set the sampling rate n such that it is sufficiently large to maintain a stable change in LIME's inherent instability, resulting in n of 4.5K, 2.5K and 1.5K for IMDB, S2D and GB dataset.

Our values for the RBO weighting were 0.75, 0.49, and 0.32 respectively for the IMDB, S2D, and GB datasets. These values were determined based on the attributes in Sec. 5.1 and the average length of the explanation features. Shorter document lengths generate smaller explanations, and so have more importance associated with the smaller number of features. We must treat altering the positions of the top 5 features in an explanation with 7 total features as more significant than for an explanation with 150 features. This significance is controlled by the RBO value with a smaller value assigning more importance to the top features.

Our values for the sampling rates were 4,500, 2,500, and 1,500 respective for the IMDB, S2D, and GB datasets. These values were derived based on the analysis shown in Fig. 3.1 where the baselines for the inherent stability were determined. Our goal is to balance explanation fidelity and running time, as is seen in Table 8.1 the computational expenditure for this process is extensive.

The maximum perturbation budget was set to 20% of the total document length (rounded up as to guarantee at least one perturbation). This budget is lower than some other work in adversarial explanations, but the requirement for consistency in the meaning of the perturbed document necessitated a modest limit to the perturbations. This consistency, especially for larger documents, began to degrade appreciably beyond $\sim$20%.

Hardware

Facts and figure relating to computation time were generated using a single NVIDIA RTX A4000 on a virtual workstation with 45 GiB RAM.

Frameworks / Software

We use an off-the-shelf LIME implementation, TEXTEXPLAINER, itself a subset of the popular explanability package ELI5*. TEXTEXPLAINER provides extensive functionality for formatting and printing explanation output. XAIFOOLER is implemented using the TEXTATTACK framework [21], a full featured package designed for adversarial attacks in natural language. TEXTATTACK's design offers the ability to quickly extend or implement most aspects of the adversarial example generation workflow.

*https://eli5.readthedocs.io/

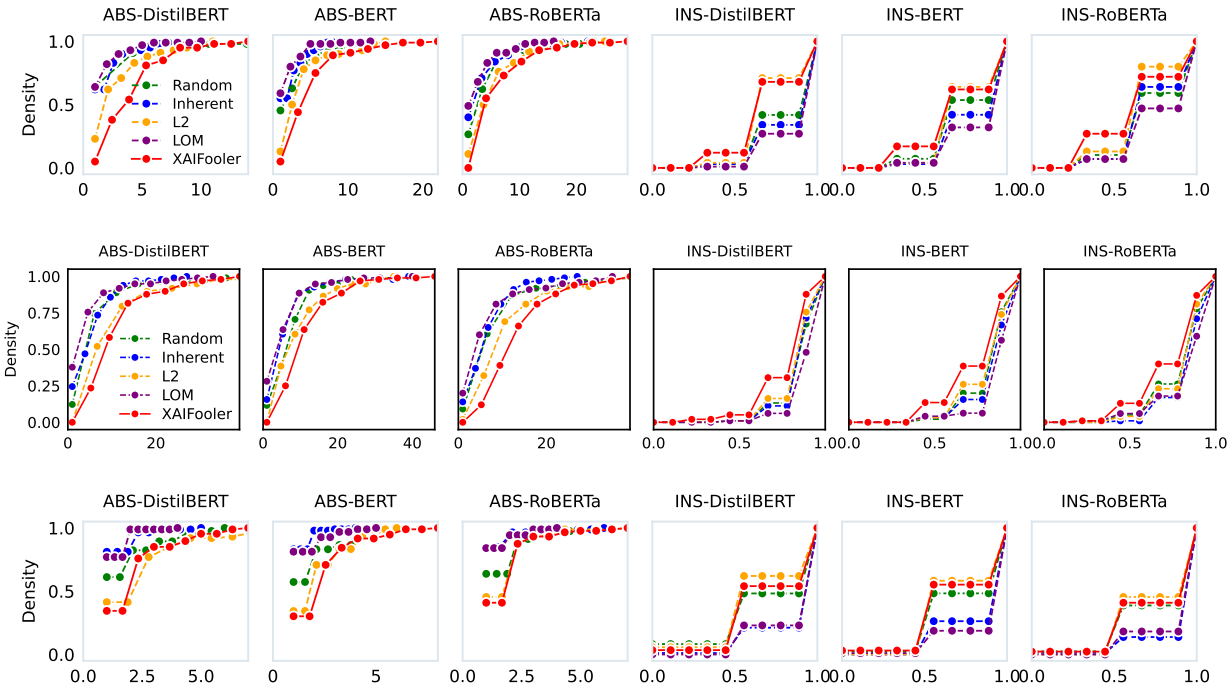


Figure 7.1: CDF plots of absolute change (ABS↑) and k-Intersection (INS↓) statistics the S2D (top), IMDB (middle) and GB (bottom) datasets.

7.3 Results

Overall

LIME is unstable and vulnerable against text perturbations. Table 7.3 shows that XAIFOOLER significantly outperformed all baselines by large margins in its ability to manipulate LIME's explanations, showing average changes in ΔABS↑, ΔRC↑ and Δ*INS*↓ of +128%, +101% and -14.63% compared to *Inherent Instability*, and of +22%, +16% and -3.15% compared to the 2nd best L_p

Method	**Average across all Results**				
	ABS↑	**SM↑**	**INS↓**	**SIM↑**	**PPL↓**
Rnd Order+Rnd Search	3.51	0.40	0.79	0.85	259.23
Rnd Order+Greedy Search	4.63	0.51	0.79	**0.91**	**123.45**
Greedy Order+Greedy Search	**6.11**	**0.59**	**0.73**	0.89	157.15

"Greedy Order+Greedy Search": Ours; "Rnd": Random

Table 7.2: Both step 1 (Greedy Order) and step 2 (Greedy Search) (Sec. 6.1) are crucial to XAIFOOLER.

baseline. Moreover, it also delivered competitive results for semantic preservation, consistently outperforming all the baselines except *LOM*, which showed to be very good at preserving the original semantics. This happened because LOM scores often reached 0.5–i.e., moving the location of centered mass to over 50% of total length, very quickly without many perturbations, yet this did not always guarantee actual changes in feature rankings.

Ablation Test

Table 7.3 confirms the appropriateness of using RBO over other similarity function such as L_p and *LOM* (Sec. 5.2). Evidently, replacing any of the procedure steps of XAIFOOLER with a *random* mechanism dropped its performance (Table 7.2). This demonstrates the importance of both Step 1 (greedy ordering of tokens to decide which one to perturb first) and Step 2 (greedy search for best perturbations) described in Sec. 6.1.

Case Study: XAI and Content Moderation

XAI systems are often used in conjunction with human-in-the-loop tasks such as online content moderation–e.g., detecting hate speech or fake news, where quality assurance is paramount. Thus, manipulations on such XAI systems will lead to human experts receiving incorrect signals and hence resulting in sub-optimal results or even serious consequences. Evaluation of XAIFOOLER on 300 positive examples from a Twitter hate speech detection dataset with $k{\leftarrow}2$ shows that XAIFOOLER often demotes the 1st-rank feature out of top-2, reducing *RC* to 0.4 and INS to 0.78, while still maintaining the original semantics with *SIM*>0.9. This means XAIFOOLER can force the AI system to output *correct predictions but for wrong* reasons. Table 7.4 describes such a case where XAIFOOLER uses the existing “thinking” to replace “think", promoting this feature to 1st rank while demoting more relevant features such as “liberals” and “israeli”.

Dataset/Method		DistilBERT					BERT					RoBERTa				
		ABS↑	RC↑	INS↓	SIM↑	PPL↓	ABS↑	RC↑	INS↓	SIM↑	PPL↓	ABS↑	RC↑	INS↓	SIM↑	PPL↓
IMDB	Inherency	4.92	0.34	0.83	1.00	33.17	5.66	0.48	0.83	1.00	34.51	5.90	0.52	0.82	1.00	33.88
	Random	5.90	0.50	0.84	0.89	71.07	6.80	0.58	0.80	0.88	75.74	7.33	0.52	0.78	0.89	73.89
	LOM	3.94	0.33	0.89	**0.96**	**40.35**	4.92	0.41	0.87	**0.95**	**43.85**	6.08	0.47	0.83	**0.95**	**43.41**
	L_p	9.07	0.52	0.81	0.89	69.54	8.76	0.57	0.79	0.89	71.16	9.96	0.65	0.78	0.89	72.87
	XAIFOOLER	**10.43**	**0.65**	**0.75**	0.89	67.01	**11.44**	**0.69**	**0.72**	0.89	67.79	**12.65**	**0.76**	**0.72**	0.90	63.15
S2D	Inherency	1.30	0.23	0.88	1.0	12.3	1.65	0.24	0.85	1.0	12.30	3.09	0.36	0.76	1.00	12.30
	Random	1.72	0.27	0.85	0.84	77.69	2.49	0.41	0.80	0.85	79.22	3.66	0.48	0.77	0.84	83.04
	LOM	1.18	0.20	0.91	**0.95**	**19.59**	1.38	0.29	0.88	**0.94**	**19.88**	2.44	0.30	0.82	**0.94**	**20.04**
	L_p	2.98	0.52	0.75	0.85	82.90	3.76	**0.54**	0.77	0.84	97.81	4.99	0.52	0.69	0.85	66.98
	XAIFOOLER	**3.95**	**0.62**	**0.73**	0.88	47.49	**4.75**	**0.54**	**0.74**	0.89	48.76	**6.11**	**0.62**	**0.67**	0.89	38.79
GB	Inherency	0.53	0.16	0.89	1.00	167.35	0.55	0.15	0.87	1.00	171.88	0.44	0.16	0.93	1.00	169.19
	Random	1.31	0.32	0.72	0.81	618.18	1.38	0.30	0.75	0.81	616.60	0.99	0.23	0.79	0.82	637.65
	LOM	0.60	0.22	0.89	**0.91**	**322.85**	0.55	0.11	0.90	**0.91**	**312.81**	0.47	0.10	0.91	**0.91**	**295.33**
	L_p	**2.06**	0.39	**0.66**	0.86	583.15	1.99	0.47	**0.71**	0.86	547.91	1.47	0.43	**0.77**	0.87	553.80
	XAIFOOLER	2.02	**0.48**	0.71	0.89	358.88	**2.10**	**0.52**	**0.71**	0.89	368.53	**1.56**	**0.45**	0.78	**0.91**	353.98

bold and underline statistics denote the best and second best results except "Inherency"

Table 7.3: Experiment results in terms of explanation changes (ABS, RC, INS) and semantic preservation (SIM, PPL)

Ranking Changes Before and After Perturbation
"@user sick of liberals thinking it's ok to dictate where they ~~think~~ **thinking** israeli jews should be allowed to live, including in israel"
liberals: 1st → 2nd; *israeli*: 3rd → 4rd; *thinking*: 5th → 1st;

Table 7.4: Case Study: Perturbation Examples and the Corresponding Changes in Explainable Feature Rankings on Twitter Hate Speech Detection

8 DISCUSSION

8.1 Discussion

Inter-dataset Instability

Fig. 7.1 illustrates the distributions of explanation shifts after perturbations on IMDB (S2D, GB results are included in the Appendix). This shows that LIME's instability expresses differently among documents, with some are easier to manipulate than others. Fig. 7.1 also shows that if one to calculate attack success rate using a threshold, which is often customized to specific applications, XAIFOOLER will still consistently outperform the baselines.

Runtime

The majority of computation time is spent generating explanations using LIME. For generating an explanation, the two key attributes are the length of the document, and the sampling rate. Table 8.1 shows that the average runtime slowly increases as the document length grows while the time per token decreases. Moreover, larger documents have more possibilities for effective perturbations, resulting in a more efficient search and ultimately fewer explanations to generate. Shorter documents are faster to run LIME, but require extensive searching to find acceptable perturbations and so generate many explanations. Furthermore, for shorter documents the explanations are quite

Model/	**DistilBERT**		**BERT**		**RoBERTA**	
Dataset	**Doc**	**Token**	**Doc**	**Token**	**Doc**	**Token**
IMDB	331.5	9.1	505.2	13.9	564.9	15.5
S2D	242.5	12.0	518.5	25.7	482.1	23.9
GB	188.7	18.2	385.7	36.9	421.1	40.6

Table 8.1: Average attack runtimes in seconds per document and token (NVIDIA RTX A4000).

stable even at rates much lower than the default, and as the document size increases, lower sampling rate values begin to degrade stability (Fig. 3.1, Table 3.1).

8.2 Future Work

While we have confirmed the lack of stability within LIME, the most pertinent question that remains is just what induces the instability. The immediately apparent sections for investigation would be the sampling procedure itself, the type of local approximation model, or the underlying model that LIME is attempting to explain. We can also recast our original question, can two similar documents produce significantly different explanations, into the related, can we produce two significantly different documents that result in the same explanation. Additionally, the question of what are the appropriate factors that determine a quality measure for the comparison of text based explanations merits interest. While RBO is an initial step towards using all the available information within an explanation, but a measure more specific to the task may prove superior. In particular, adjusting the weighting scheme of RBO using some function of the explanation weights.

8.3 Limitations

Our method and the results collected from it used the default settings and parameters for the explanation generation framework within TEXTEXPLAINER* with the exception of the sampling rate n. This departure is justified by the results shown in Table 3.1 and especially Fig. 3.1, where we see a rapid diminishing returns in explanation fidelity as the sample rate n increases. Although our choice of sampling rates are already significantly higher than prior works, this reduction in sampling rate was necessary due to the extensive computation time required, with the explanation generating process being the largest proportion of computation time. The extensive computation time prevented extensive testing with a large range of sampling rates, especially those higher than the default of $n{=}5K$. We leave the question of what effect, if any, "oversampling" with rates much higher than the default has on stability to future work. It remains unexplored just how much effect

*`https://eli5.readthedocs.io/`

the base model itself has on the resulting explanation similarity. That is, are certain type of models amplifying LIME's unstable results? This question also applies to the LIME itself. Our choice of surrogate model was the default, but this can be replaced. What are the differences between the choices for local model when it comes to stability?

8.4 Conclusion

This paper affirms that LIME is inherently unstable and its explanations can be exploited via text perturbations. We re-purpose and apply RBO as a key part of our algorithm XAIFOOLER, which we have shown to provide competitive results against other adversarial explanation generation methods, and do so in a way that satisfies our criteria from earlier when determining just how best to compare an explanation. Future works include understanding the interactions between the target model and LIME that lead to such instability.

BIBLIOGRAPHY

[1] David Alvarez-Melis and Tommi S Jaakkola, *On the robustness of interpretability methods*, ICML Workshop on Human Interpretability in Machine Learning (WHI 2018) (2018).

[2] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al., *Universal sentence encoder*, arXiv preprint arXiv:1803.11175 (2018).

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, *BERT: pre-training of deep bidirectional transformers for language understandingg*, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) (2019), 4171–4186.

[4] Emily Dinan, Angela Fan, Ledell Wu, Jason Weston, Douwe Kiela, and Adina Williams, *Multi-dimensional gender bias classification*, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP) (Online), Association for Computational Linguistics, November 2020, pp. 314–331.

[5] Finale Doshi-Velez and Been Kim, *Towards a rigorous science of interpretable machine learning*, arXiv preprint arXiv:1702.08608 (2017).

[6] Jordan D Fuhrman, Naveena Gorre, Qiyuan Hu, Hui Li, Issam El Naqa, and Maryellen L Giger, *A review of explainable and interpretable ai with applications in covid-19 imaging*, Medical Physics **49** (2022), no. 1, 1–14.

[7] Damien Garreau and Ulrike Luxburg, *Explaining the explainer: A first theoretical analysis of lime*, International Conference on Artificial Intelligence and Statistics, PMLR, 2020, pp. 1287–1296.

[8] Damien Garreau and Ulrike von Luxburg, *Looking deeper into tabular lime*, arXiv preprint arXiv:2008.11092 (2020).

[9] Amirata Ghorbani, Abubakar Abid, and James Zou, *Interpretation of neural networks is fragile*, Proceedings of the AAAI conference on artificial intelligence, vol. 33, 2019, pp. 3681–3688.

[10] Alex Gramegna and Paolo Giudici, *Shap and lime: an evaluation of discriminative power in credit risk*, Frontiers in Artificial Intelligence **4** (2021), 752558.

[11] Adam Ivankay, Ivan Girardi, Chiara Marchiori, and Pascal Frossard, *Fooling explanations in text classifiers*, International Conference on Learning Representations (ICLR) (2022).

[12] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits, *Is bert really robust? a strong baseline for natural language attack on text classification and entailment*, Proceedings of the AAAI conference on artificial intelligence, vol. 34, 2020, pp. 8018–8025.

[13] Ravi Kumar and Sergei Vassilvitskii, *Generalized distances between rankings*, Proceedings of the 19th International Conference on World Wide Web (New York, NY, USA), WWW '10, Association for Computing Machinery, 2010, p. 571–580.

[14] Nesaretnam Barr Kumarakulasinghe, Tobias Blomberg, Jintai Liu, Alexandra Saraiva Leao, and Panagiotis Papapetrou, *Evaluating local interpretable model-agnostic explanations on clinical machine learning classification models*, 2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS), IEEE, 2020, pp. 7–12.

[15] Piyawat Lertvittayakumjorn and Francesca Toni, *Explanation-based human debugging of nlp models: A survey*, Transactions of the Association for Computational Linguistics **9** (2021), 1508–1528.

[16] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, *Roberta: A robustly optimized bert pretraining approach*, arXiv preprint arXiv:1907.11692 (2019).

[17] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts, *Learning word vectors for sentiment analysis*, Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Portland, Oregon, USA), Association for Computational Linguistics, June 2011, pp. 142–150.

[18] Dina Mardaoui and Damien Garreau, *An analysis of lime for text data*, International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 3493–3501.

[19] Aniek F. Markus, Jan A. Kors, and Peter R. Rijnbeek, *The role of explainability in creating trustworthy artificial intelligence for health care: A comprehensive survey of the terminology, design choices, and evaluation strategies*, Journal of Biomedical Informatics **113** (2021), 103655.

[20] Christoph Molnar, *Interpretable machine learning*, https://christophm.github.io/interpretable-ml-book/, 2018, `https://christophm.github.io/interpretable-ml-book/`.

[21] John X. Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi, *Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp*, 2020.

[22] Tien N Nguyen and Raymond Choo, *Human-in-the-loop xai-enabled vulnerability detection, investigation, and mitigation*, 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), IEEE, 2021, pp. 1210–1212.

[23] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar, *A survey on deep learning: Algorithms, techniques, and applications*, ACM Computing Surveys (CSUR) **51** (2018), no. 5, 1–36.

[24] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, *Language models are unsupervised multitask learners*, (2019).

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

[26] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf, *Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter*, arXiv preprint arXiv:1910.01108 (2019).

[27] Sanchit Sinha, Hanjie Chen, Arshdeep Sekhon, Yangfeng Ji, and Yanjun Qi, *Perturbing inputs for fragile interpretations in deep natural language processing*, arXiv preprint arXiv:2108.04990 (2021).

[28] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju, *Fooling lime and shap: Adversarial attacks on post hoc explanation methods*, Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, 2020, pp. 180–186.

[29] Ilia Stepin, Jose M Alonso, Alejandro Catala, and Martín Pereira-Fariña, *A survey of contrastive and counterfactual explanation generation methods for explainable artificial intelligence*, IEEE Access **9** (2021), 11974–12001.

[30] Erico Tjoa and Cuntai Guan, *A survey on explainable artificial intelligence (xai): Toward medical xai*, IEEE transactions on neural networks and learning systems **32** (2020), no. 11, 4793–4813.

[31] William Webber, Alistair Moffat, and Justin Zobel, *A similarity measure for indefinite rankings*, ACM Transactions on Information Systems (TOIS) **28** (2010), no. 4, 1–38.

[32] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu, *From paraphrase database to compositional paraphrase model and back*, Transactions of the Association for Computational Linguistics **3** (2015), 345–358.

[33] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang, *Interpretable deep learning under fire*, 29th USENIX security symposium (USENIX security 20), 2020.

VITA

Christopher Burger attended Mansfield University of Pennsylvania and earned a Bachelor of Science (2018) with majors in Applied Mathematics, Computational Mathematics, and Pure Mathematics. He is currently enrolled in the PhD program in the Department of Computer and Information Science at the University of Mississippi.