

Does It Make Sense to Explain a Black Box With Another Black Box?

Julien Delaunay¹, Luis Galárraga¹, and Christine Largouët²

¹ INRIA/IRISA, Rennes, France {julien.delaunay, luis.galarraga}@inria.fr

² L'Institut Agro/IRISA, Rennes, France christine.largouet@irisa.fr

Abstract. Although counterfactual explanations are a popular approach to explain ML black-box classifiers, they are less widespread in NLP. Most methods find those explanations by iteratively perturbing the target document until it is classified differently by the black box. We identify two main families of counterfactual explanation methods in the literature, namely, (a) *transparent* methods that perturb the target by adding, removing, or replacing words, and (b) *opaque* approaches that project the target document into a latent, non-interpretable space where the perturbation is carried out subsequently. This article offers a comparative study of the performance of these two families of methods on three classical NLP tasks. Our empirical evidence shows that opaque approaches can be an overkill for downstream applications such as fake news detection or sentiment analysis since they add an additional level of complexity with no significant performance gain. These observations motivate our discussion, which raises the question of whether it makes sense to explain a black box using another black box.³

Keywords: Interpretability · Natural Language Processing · Counterfactual Explanations.

1 Introduction

The latest advances in machine learning (ML) have led to significant advances in various natural language processing (NLP) tasks [3, 13, 25], such as text generation, fake news detection, sentiment analysis, and spam detection. These notable improvements can be partly attributed to the adoption of methods that encode and manipulate text data using latent representations. Those methods embed text into high-dimensional vector spaces that capture the underlying semantics and structure of language, and that are suitable for complex ML models.

Despite the impressive gains in accuracy achieved by modern ML algorithms [2, 3], their utility can be diminished by their lack of interpretability [26]. This has, in turn, raised an increasing interest in ML explainability, the task of providing appropriate explanations for the answers of black-box ML algorithms [10]. Indeed, a model could make correct predictions for the wrong reasons [7, 16].

³ This article was originally published in French at the Journal TAL. VOL 64 n°3/2023

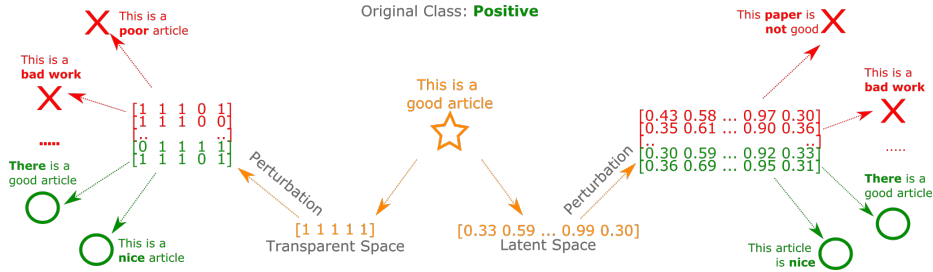


Fig. 1: The mechanism employed to perturb the target documents by the transparent and opaque methods. Transparent techniques, on the left, convert the input text to a vector representation, where ‘1’ indicates the presence of the input word and ‘0’ denotes a replacement. Opaque methods, as on the right, embed words from the target text into a latent space and perturb the text in this high-dimensional space.

Unless the ML model is a white box, explaining the results of such an agent requires an explanation layer that elucidates the internal workings of the black box in a post-hoc manner.

While there are several ways to explain the outcomes of an ML model a posteriori, there has been a growing emphasis on counterfactual explanations, a domain that has experienced notable popularity over the last five years [6, 17]. A counterfactual explanation is a counter-example that is similar to the original text, but that elicits a different outcome in the black box [28]. Consider the classifier depicted in Figure 1, for sentiment analysis applied to the review “This is a good article” – classified as positive. In this toy example, a counterfactual could be the phrase “This is a **poor** article”. This explanation tells us that the adjective “good” was a possible reason for this sentence to be classified as positive, and changing the polarity of that adjective changes the classifier’s response.

In the literature, counterfactual explanation methods operate by increasingly perturbing the target text until the answer from the model – often a classifier – changes. Those perturbations can be conducted *transparently* by adding, removing, or changing words and syntactic groups [15, 22, 31] in the original target text as depicted in Figure 1. Since removing or adding words from a text can lead to unrealistic texts, more recent methods [8, 21, 24] embed the target text in a latent space that captures the underlying distribution of the model’s training corpus. Perturbations are then carried out in this space and then brought back to the space of words to guarantee realistic counterfactual explanations. These explanation methods rely on *opaque* sophisticated techniques to compute those explanations [12], which is tantamount to explaining a black box with another black box.

Based on this somehow paradoxical observation, we conduct a comparative study of various transparent and opaque post-hoc counterfactual explanation

approaches. Rather than two distinct categories, the studied methods define a continuum, as some methods may combine transparent and non-interpretable techniques. Our empirical analyses have revealed that, for certain NLP tasks such as spam detection, fake news detection, or sentiment analysis, learning a compressed representation may be unnecessary. To illustrate this point and as a proof of concept, we have developed two transparent counterfactual explanation techniques that outperform opaque methods. This is largely explained by the fact that opaque approaches often generate non-intuitive counterfactual explanations, *i.e.*, counterexamples that bear no resemblance to the target text. This approach goes against not only the nature of counterfactual explanations but also raises questions about the true level of transparency achieved when explaining one black box with another.

Thus, the key contributions of this paper are as follows:

1. The proposal of a spectrum that evaluates the complexity of counterfactual explanations, and provides a nuanced perspective on these methods.
2. A comparative study of different counterfactual methods, each representing a part of the spectrum.

The document is structured as follows. Section 2 defines transparent and opaque counterfactual explanation methods. Then, Section 3 reviews the state of the art. Section 4 introduces two new transparent methods, which we then analyze in light of the spectrum of existing transparent and opaque techniques (Section 5). We then detail the experimental protocol of our comparative study in Section 6. The results of our experiments are presented in Section 7. Section 8 concludes the paper with a discussion of our findings and the limitations of our study.

2 Transparent vs. Opaque Methods

In the introduction we categorized counterfactual explanation techniques as either opaque or transparent. We now define these notions in a formal way.

2.1 Transparent Methods

Implicitly, transparent counterfactual methods model a text $x \in X$ of length d with words from a vocabulary Σ , as a binary sparse matrix of dimension $|\Sigma| \times d$. Here $x_{ij} = 1$ means that the i -th word of the vocabulary Σ occurs at position j -th position in x . A perturbed text z is then obtained as an additive perturbation

$$z = X + \epsilon, \quad \text{with} \quad z_{ij} = \max(0, \min(1, x_{ij} + \epsilon_{ij})),$$

where ϵ is a noise matrix such that ϵ_{ij} is restricted to three values: -1 for removing word $i \in [1, \dots, |\Sigma|]$ at position $j \in [1, \dots, d]$, 0 for doing nothing, and 1 for adding word i at position j . The clipping operation $\max(0, \min(1, \cdot))$ makes sure that z is also a binary matrix.

2.2 Opaque Methods

Opaque methods generate counterfactual candidates z by adding noise to the representation of the target text $x \in X$ in a latent space. If we denote such a representation by $g(x)$, this is expressed as $z = g^{-1}(g(x) + \epsilon)$, where $g : X \rightarrow \mathbb{R}^{d'}$ is a transformation function into a latent space in $\mathbb{R}^{d'}$ (for some hyperparameter d'), and $\epsilon \in \mathbb{R}^{d'}$ is a noise vector. Opaque methods must also define the inverse function g^{-1} that maps a vector of real numbers into a text.

3 Related Works

Counterfactual explanation methods compute contrastive explanations for ML black-box algorithms by providing examples that resemble a target instance but that lead to a different answer in the black box [28]. These counterfactual explanations convey the minimum changes in the input that would modify a classifier’s outcome. Social sciences [17] have shown that human explanations are contrastive and Wachter et al. [28] have illustrated the utility of counterfactual instances in computational law. When it comes to NLP tasks, a good counterfactual explanation should be fluent [30], *i.e.*, read like something someone would say, and be sparse [27], *i.e.*, look like the target instance.

Counterfactual approaches have gained popularity in the last few years. As illustrated by the surveys, first by Bodria et al. [1] and later by Guidotti [6], around 50 additional counterfactual methods appeared in a one-year time span. Despite this surge of interest in counterfactual explanations, their study for NLP applications remains underdeveloped [22]. In the following, we elaborate on the existing counterfactual explanation methods for textual data along a spectrum that spans from transparent to opaque approaches.

Transparent Approaches. One of the first transparent counterfactual explanation approaches was proposed by Martens and Provost [15] who introduced Search for Explanations for Document Classification (SEDC), a method that removes the words for which the classifier exhibits the highest *sensitivity*. These are words that impact the classifier’s prediction the most. More recently, Ross et al. [22] developed Minimal Contrastive Editing (MICE), a method that employs a Text-To-Text Transfer Transformer to fill masked sentences. In [31], the authors presented Plausible Counterfactual Instances Generation (PCIG), which generates grammatically plausible counterfactuals through edits of single words with lexicons manually selected from the economics domain.

Opaque Methods. Methods such as Decision Boundary [8], xSPELLS [24] or counterfactualGAN (cfGAN) [21] operate in three phases. First, they embed the target text onto a latent space. This is accomplished by employing specific techniques such as Variational AutoEncoder (VAE) in the case of xSPELLS, or a pre-trained language model (LM) for cfGAN. Second, while the classifier’s decision boundary is not traversed, these methods perturb the latent representation of the target phrase. This is done by adding Gaussian noise in the case of xSPELLS, whereas cfGAN resorts to a Conditional Generative Adversarial Net-

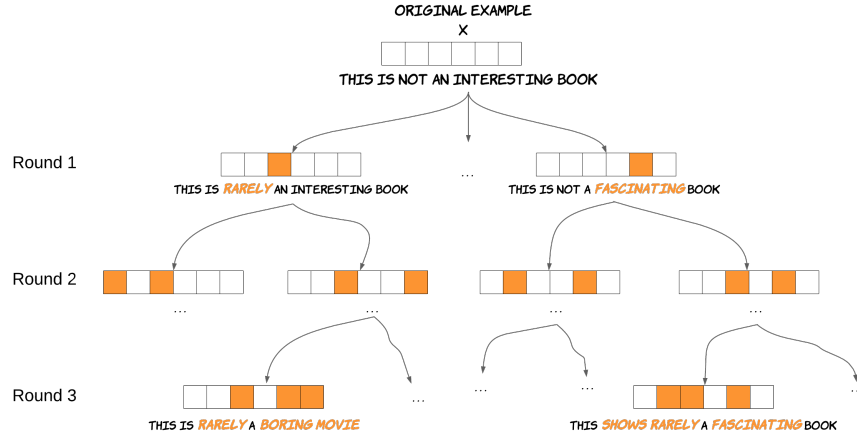


Fig. 2: The tree structure of the algorithm used to iteratively perturb the target document. At each round, a word from the target text is iteratively replaced by a word from its corresponding set of potential replacement words. Thus, with each successive round, the number of word replacements for generating artificial documents increases.

work. Finally, a decoding stage produces sentences from the latent representation of the perturbed documents.

There also exist methods such as Polyjuice [30], Generate Your Counterfactuals (GYC) [14] and Tailor [23] that perturb text documents in a latent space, but can be instructed to change particular linguistic aspects of the target text, such as locality or grammar tense. Such methods are not particularly designed to compute counterfactual explanations but are rather conceived for other applications such as data augmentation.

Unlike pure word-based perturbation methods, latent representations are good at preserving *semantic closeness* for small perturbations. That said, these methods are not free of pitfalls. First, methods such as xSPELLS and cfGAN are deemed opaque since a latent space is not human-understandable [26]. Moreover, existing latent-based approaches do not seem optimized for sparse counterfactual explanations – one of the defining features of a counterfactual. We show this through our experimental results that suggest that a minor alteration in the latent space can cause a significant alteration in the original space.

4 Counterfactual Explanation Methods for Textual Data

Before elaborating on our study, we introduce two novel counterfactual explanation techniques, aimed to enrich the middle ground between fully opaque and fully transparent approaches. The methods are called Growing Language and

Growing Net, and both depend on an iterative process that replaces words within a target text $x = (x_1, \dots, x_d) \in X$ ($x_i \in \Sigma$ are words from a vocabulary Σ) until the predicted class of a given classifier $f : X \rightarrow Y$ changes. The goal of such a procedure is to compute sparse counterfactual explanations with the fewest modified words.

Algorithm 1 Explore

Require: target text $x = (x_1, \dots, x_d) \in X$,
a classifier $f : X \rightarrow Y$,
SIMWORDS($\cdot$) $\rightarrow$ retrieves similar words
Hyper-parameters: $n = 2000$

Ensure: one or multiple counterfactual instances

```

1: Initialise  $W = (W_1, \dots, W_d)$ , sets of candidate words
2: for  $i \leftarrow 1$  to  $d$  do
3:    $W_i \leftarrow \text{SIMWORDS}(x_i, \text{POS}(x_i))$ 
4: end for
5: Initialize  $Z = (z_1, \dots, z_n)$  as  $n$  copies of  $x$ 
6: Initialize  $C \leftarrow \emptyset$ ;  $r \leftarrow 0$ 
7: while  $r < d \wedge C = \emptyset$  do
8:    $r \leftarrow r + 1$ 
9:   for  $j \leftarrow 1$  to  $n$  do ▷ For each copy of  $x$ 
10:    for  $l \leftarrow 1$  to  $r$  do
11:       $k \leftarrow \text{random}(0, d)$  ▷  $k : z_j^k = x_k$ 
12:       $z_j^k \leftarrow \text{random word from } W_k$ 
13:    end for
14:    if  $f(x) \neq f(z_j)$  then
15:       $C \leftarrow C \cup \{z_j\}$ 
16:    end if
17:  end for
18: end while
19: return  $C$ 

```

Algorithm 1 outlines the iterative exploration process employed by Growing Language and Growing Net. In the first step (lines 1 to 4), both approaches generate d sets of potential word replacements $W_1, \dots, W_d$ for each word x_i in the target document x . Those replacements must have the same part-of-speech (POS) tag as x_i . The external module to obtain those word replacements depends on the method. These modules are detailed later. Subsequently, our methods create artificial documents iteratively (lines 7 and 18) and exemplified as a tree structure in Figure 2. These documents are generated while some words in the original document remain non-replaced ($r < d$), or while we have not found any counterfactuals ($C = \emptyset$). At each iteration, the exploration keeps n copies of the original text (x) on which we replace r individual words (x_k) with randomly selected words from their respective sets of potential replacements (W_k). Line 11 ensures that the replaced word comes from the original sentence ($k : z_j^k = x_k$)

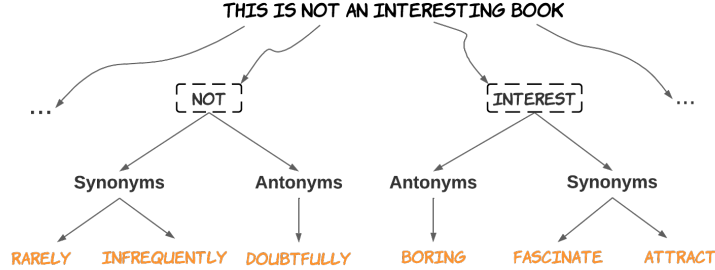


Fig. 3: Diagram representing the mechanisms of the Growing Net approach. By leveraging the tree-like structure of WordNet, Growing Net generates sets of words that can replace each term in the target document.

Algorithm 2 Growing Net

Require: a target text $x = (x_1, \dots, x_d) \in X$,
 a classifier $f : X \rightarrow Y$;
 1: $C \leftarrow \text{explore}(x, f, \text{WN_SIMWORDS}_{t=1}(\cdot))$
 2: **return** $\text{argmax}_{c \in C} \text{Wu-P}(c, x)$

and actually replaces r words instead of words already modified. Finally, lines 14-16 check if the resulting phrases are counterfactual instances.

For example, consider the target review, “*This is not an interesting book*”, classified as negative by a sentiment analysis model (Figure 2). In the first round, our routine produces artificial reviews with only one modified word. Subsequent rounds will replace two words and so on (lines 10 to 13). In this process, counterfactuals are identified, and the closest one is returned as the explanation. These methods prioritize counterfactuals closely related to the original document to provide concise and meaningful explanations.

4.1 Growing Net

This method capitalizes on the rich structure of WordNet [4] to identify potential word replacements. WordNet is a lexical database and thesaurus that organizes words and their meanings into a semantic tree of interrelated concepts. The method is described in Algorithm 2, and uses the module WN_SIMWORDS_t . In the exploration phase, Growing Net uses WN_SIMWORDS_t to find words at a distance of at most t in the WordNet hierarchy among synonyms, antonyms, hyponyms, and hypernyms for a given word x_i to replace. This process is illustrated in Figure 3. In our experiments we set $t = 1$ as this value already yields good results – higher values would incur longer runtimes. The exploration returns a set of counterfactuals, from which Growing Net selects the one with the highest Wu-Palmer Similarity (Wu-P) [29] as final explanation. This similarity score for

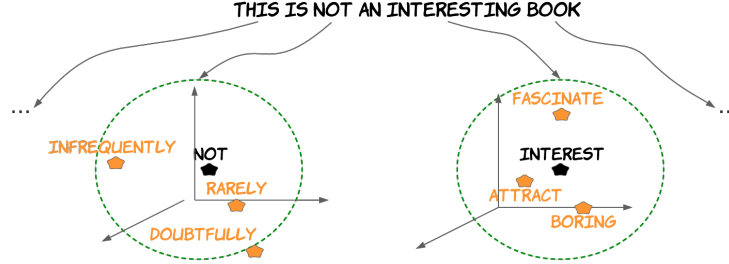


Fig. 4: Diagram illustrating the operation of the Growing Language method. The words in the target text are transformed into a latent representation using a large language model. In this latent space, words with similarities become potential replacements for generating artificial documents.

Algorithm 3 Growing Language

Require: target text $x = (x_1, \dots, x_d) \in X$
 a classifier $f : X \rightarrow Y$;
 Hyper-parameters: $\tau = 0.02$; $\theta = 0.9$; $\theta_{min} = 0.4$;

- 1: $C \leftarrow \emptyset$
- 2: **while** $\theta > \theta_{min} \wedge C = \emptyset$ **do**
- 3: $C \leftarrow C \cup \text{explore}(x, f, \text{LM_SIMWORDS}_\theta(\cdot))$
- 4: $\theta \leftarrow \theta - \tau$
- 5: **end while**
- 6: **return** $\text{argmin}_{c \in C} \|x - c\|_0$

text relies on Wordnet, and takes into account the relatedness of the concepts in the phrase, e.g., via the path length to their most common ancestor in the hierarchy.

4.2 Growing Language

This approach leverages the power of language models (LM) to restrict the space of possible word replacements via the module $\text{LM_SIMWORDS}_\theta$ (see Algorithm 3). Large language models are powerful natural language processing AI systems. They are employed in this context to embed words into numerical representations, often referred to as a latent space. In simpler terms, a latent representation consists of high-dimensional vectors that capture the underlying semantic and contextual information of the original text. Given a word x_i to replace, $\text{LM_SIMWORDS}_\theta$ embeds the word onto the latent space of an LM, as illustrated in Figure 4. Then $\text{LM_SIMWORDS}_\theta$ retrieves words whose latent representation is at a distance of θ at most. In our experiments, we initially set this threshold to 0.8 on a scale from 0 to 1. If for a given θ , Growing Language

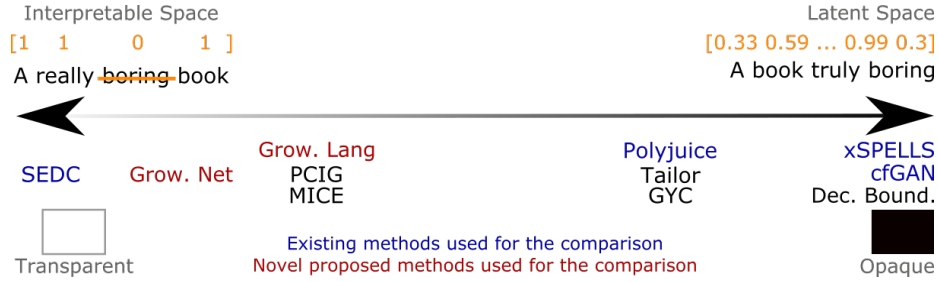


Fig. 5: Spectrum for counterfactual explanation techniques that goes from the most transparent methods on the left to the most opaque on the right. Transparent methods perturb documents in a binary space; opaque methods do it in a latent space.

cannot find counterfactual instances, the distance threshold is relaxed, i.e., reduced by τ (set to 0.02 in our experiments), so that the exploration routine considers more words (line 4 Algorithm 3). This allows Growing Language to maintain computational efficiency, as starting with a low threshold could result in longer processing times. Should multiple counterfactuals be found, Growing Language selects the one with the fewest modifications compared to the original document (minimal L0 distance). For our experiments, we employed the model `en_core_web_md` from the library Spacy [9], but any language model capable of embedding words and offering word distances could be applied in this context.

5 Interpretability Spectrum

We highlight that the categorisation transparent vs. opaque for counterfactual explanation method defines the two ends of a continuum, which we depict in Figure 5. This spectrum ranges from the most transparent methods on the left to the most opaque methods on the right. We distinguish two sub-categories of transparent methods: **fully transparent** methods and **partially transparent** methods. In the first category, individuals can understand why adding noise to a word produces a specific result, such as replacing a word with its antonym, for example. In contrast, for partially transparent methods the choice of the perturbation ϵ may resort to black-box machinery – even though perturbation itself can be modelled a sum of sparse binary matrices. We also define two sub-categories of opaque methods: **partially opaque** methods and **fully opaque** methods. In the first category, the perturbation in the latent space is still controlled by an interpretable symbol, e.g., a control code as in [30]. Conversely, it is difficult, if not impossible, to understand the semantics of the perturbation in the latent space of fully opaque methods. We situate the different methods in the state of the art across the spectrum shown in Figure 5.

Fully Transparent. At the leftmost end of the spectrum, we find the method SEDC [15], which perturbs text instances by hiding only highly sensitive words

within the text. We place Growing Net on the right of SEDC, because it goes beyond simple word masking. Instead, it substitutes words judiciously via an external interpretable asset, namely Wordnet.

Partially Transparent. Methods like PCIG [31], MICE [22], and Growing Language are considered more opaque than Growing Net, because they employ a latent space to identify semantically close word substitutions. Despite this reliance on black-box techniques, we consider them transparent because the search for counterfactuals is still carried out in the space of words.

Partially Opaque. Polyjuice, Tailor, and GYC fall in the category of partially opaque methods, as they leverage control codes to perturb the target document. Control codes are specific instructions that adapt the perturbation of the target text so that it complies with a specific task, such as translating, summarizing, or changing the tense of a text. While these modifications occur in a latent space, the inclusion of control codes provides some level of clarity regarding why a modification influences the model’s prediction.

Fully Opaque. On the far right of the interpretability spectrum, we encounter fully opaque approaches such as Decision Boundary, xSPELLS and cfGAN. These methods perturb instances in a latent space, making it challenging for users to discern the underlying process of counterfactual generation.

This interpretability spectrum provides valuable insights into the transparency and opacity of counterfactual explanation methods, allowing for a more nuanced understanding of their capabilities.

6 Experimental Protocol

Having introduced the spectrum of counterfactual explanation methods across the interpretability axis, we now describe the experimental setup designed to evaluate those methods. The code of the studied methods, the datasets, and the experimental results are available at <https://github.com/j2launay/ebbwbb>.

6.1 Methods

We picked a set of representative domain-agnostic methods from all regions of the spectrum depicted in Figure 5. These include SEDC and Growing Net among the fully transparent methods, Growing Language among the partially transparent methods, Polyjuice among the partially opaque methods, and xSPELLS and cfGAN among the group of fully opaque methods.

It is important to note that we excluded the PCIG [31] and MICE [22] methods from our further analysis, each being excluded for specific reasons. In the case of PCIG, this method relies on domain-specific rules in economics, limiting its relevance to our diverse datasets. Regarding MICE, it utilizes transformer models to identify semantically relevant word replacements, which is computationally expensive according to the authors. This complexity runs counter to our goal of favoring simpler transparent methods.

Dataset	Number of Words			Instances	Accuracy (%)		
	Total	Average	σ		MLP	RF	BERT
Fake	19419	11.8	3.2	4025	84 %	84 %	91 %
Polarity	11646	20.8	9.3	10660	72 %	67 %	82 %
Spam	15587	18.5	10.6	8559	100 %	100 %	100 %

Table 1: Information about the experimental datasets. The “average” column denotes the average number of words per instance (document). The column “ σ ” represents the standard deviation while the column “Instances” shows the number of text documents per dataset.

Additionally, we deliberately omitted adversarial methods from our analysis. These methods are designed to induce errors in model predictions rather than for explanatory purposes [18]. Thus, we excluded them to emphasize the fundamental difference in their objective compared to counterfactual explanation methods. We favored approaches focused on understanding model decisions rather than manipulating its predictions. Similarly, we removed the Linguistically-Informed Transformation (LIT) [11], a method aimed at automatically generating sets of contrasts. LIT aims to produce documents outside the data distribution, making them unrealistic and diverging from our goal of faithful and relevant explanations.

Detailed information regarding the implementation, versions, and hyperparameters of each counterfactual explanation method used in our experiments is available in Appendix A.

6.2 Tasks & Datasets

We conduct the evaluation on three popular downstream tasks: (a) spam detection in messages, (b) sentiment analysis, and (c) detection of fake news from newspaper headlines. The datasets associated to these tasks consist of two target classes, and contain between 4000 and 10660 textual documents. The average number of words in each document is between 11.8 and 20.8 as reported in Table 1. Concerning the polarity [19] and spam detection [5] datasets, we downloaded the data from Kaggle. Regarding the fake news detection dataset, we constructed it using real headlines from a dataset ⁴ combined with headlines fabricated from a fake news dataset ⁵. This combined dataset is publicly available in our repository <https://github.com/j21aunay/ebbwbb>.

6.3 Black-box Classifiers

Our evaluation uses two distinct black-box classifiers implemented using the scikit-learn library and already employed in [24]. These black boxes are (i) a

⁴ <https://www.kaggle.com/datasets/rmisra/news-category-dataset>

⁵ <https://www.kaggle.com/competitions/fake-news/overview>

Random Forest (RF) consisting of 500 tree estimators, (ii) a multi-layer perceptron (MLP) with token counts as input, and (iii) a classifier based on Distill-BERT with a sequence classification head on top⁶. For the RF and the MLP, we employed both the *token count* and *tf-idf* vectorizers to convert text into proper inputs for the models.

We used 70% of the instances for training, and the remaining for testing. We also selected the target instance to explain within this test set. Across all datasets, the average accuracy of these three classifiers ranges from 67% to 100%. Detailed results are presented in Table 1.

All the experiments were run in a server with an Intel(R) Xeon(R) Gold 5220 CPU (2.20GHz, 18 cores, 24MB L3 cache) and 96GB of RAM (DDR4).

7 Results

We now present the results of our evaluation, organized in four rounds of experiments categorized according to two aspects. First, we assess the quality of the produced counterfactual explanations based on two essential criteria: (i) **minimality**, and (ii) **plausibility**. Second, we evaluate the methods themselves in terms of (iii) **flip change**, and (iv) **runtime**. For each evaluated method and black-box classifier, we computed counterfactual explanations for 100 target texts extracted from the test sets of our datasets.

7.1 Counterfactual Quality

A high-quality textual counterfactual explanation tells us what are the most sensitive parts or aspects of the target phrase, that otherwise changed, would lead to a different classification outcome. As we mentioned in Section 3, it follows then that such an explanation must (i) incur minimal changes w.r.t the target phrase (sparse changes), and (ii) be linguistically plausible, i.e., sound like something a person would naturally write or say [6].

Minimality. We quantify the minimality criterion by measuring the distance between the counterfactual and the target sentence. Figure 6 and 7 display the results of our minimality assessments, considering both the Levenshtein distance and the cosine similarity within the embedding space of the BERT-Sentence model [20]. This dual approach ensures a comprehensive evaluation, accounting for both lexical similarity and latent features, including aspects of style.

Notably, our findings reveal that methods positioned in the middle-ground, particularly Growing Net, performed favorably compared to opaque approaches, both in terms of the number of words modified and semantic comparison. It is worth noting that xSPELLS introduced the most significant changes to the original text – contradicting one of the main functional requirements of a counterfactual explanation [28]. Similarly, we observe a high variance in the minimality of

⁶ <https://is.gd/zljJJN>

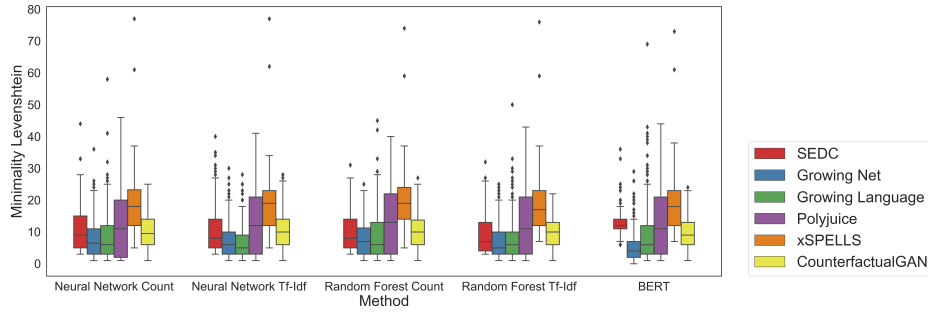


Fig. 6: Minimality as the levenshtein edit distance between the closest counterfactual and the target text ($\downarrow$ better).

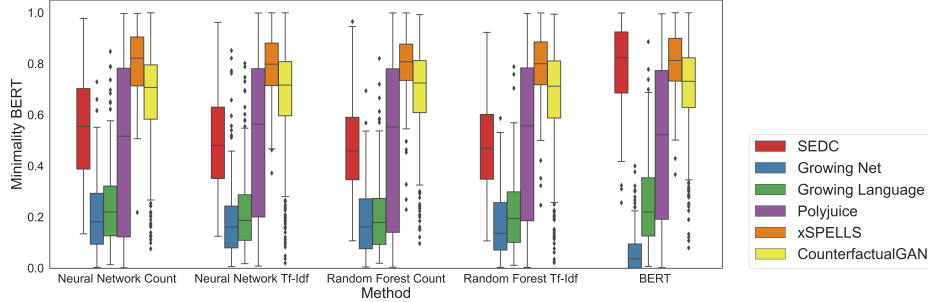


Fig. 7: Minimality as the Sentence-BERT embedding distance between the closest counterfactual and the target text ($\downarrow$ better).

the counterfactuals generated by Polyjuice, indicating that some counterfactuals were notably distant from their corresponding target instances. While these methods introduced minor perturbations to the original text, these modifications occurred within a latent space. Nothing guarantees, however, that these minor adjustments translate into visually subtle modifications of the target phrase when the resulting phrase is brought back to the original space. As an example, consider the target text “This is one of Polanski’s best films.” from the polarity dataset. For the DistillBERT classifier, cfGAN returns the counterfactual “this is one of **shot kingdom intelligence’ s all**”, which looks completely unrelated to the target text. Conversely, the transparent method SEDC produces the counterfactual “This is one of **MASK MASK MASK**”, whereas Growing Language outputs “This is one of Polanski’s **worst** films.”. For more examples of counterfactuals generated by each method, please see appendix B.

Additionally, we noted first that when the complexity of the classifier increases, the counterfactual explanations generated by SEDC lie farther from the original text. Secondly, we observe minor variations dependent on the vectorizer employed by the classifiers (*count* or *tf-idf*). Hence, for the subsequent phase of the evaluation, we present results exclusively for the *tf-idf* vectorizer.

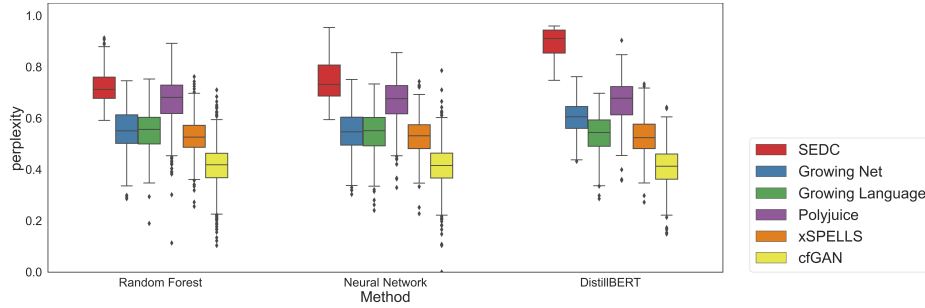


Fig. 8: Perplexity as the MSE loss of a GPT model on the generated counterfactuals ($\downarrow$ better).

Plausibility. While linguistic plausibility is typically evaluated through user studies [14, 30], we approximate it here following the techniques from Ross et al. [22, 23]. Thus, we use perplexity scores based on a GPT language model (GPT-3) [2], by calculating the average mean squared error (MSE) loss when predicting the next word in the counterfactual given the preceding words. Figure 8 presents the plausibility of the counterfactuals. To enhance comparability, we normalized perplexity scores based on the maximum perplexity observed across the entire set of counterfactuals, where lower scores indicate higher plausibility. Notably, SEDC and Polyjuice generated texts with the lowest plausibility, which is expected since SEDC masks words, leading sometimes to nonsensical sentences. In contrast, cfGAN demonstrated the highest plausibility, while both Growing Net and Language achieved perplexity scores similar to those of xSPELLS.

Dataset	Fake			Spam			Polarity		
	MLP	RF	BERT	MLP	RF	BERT	MLP	RF	BERT
SEDC	0.95	0.82	1	0.47	0.42	0.56	0.92	0.93	0.98
Grow. Net	0.90	0.8	0.88	0.44	0.29	0.84	0.97	0.98	0.90
Grow. Lang.	0.84	0.84	0.77	0.58	0.61	0.17	0.92	0.92	0.92
Polyjuice	0.26	0.23	0.21	0.17	0.14	0.16	0.33	0.31	0.29
xSPELLS	0.68	0.78	0.77	0.98	0.95	0.91	0.91	0.76	0.91
cfGAN	0.18	0.12	0.09	0.14	0.05	0.03	0.50	0.50	0.48

Table 2: Average label flip per dataset and black box of the six counterfactual methods ($\uparrow$ better).

7.2 Method Quality

We now compare the quality of the counterfactual explanation methods themselves based on (iii) label flip rate, which measures how frequently a method produces an instance classified differently by the model, and (iv) runtime, the time it takes for each method to generate a counterfactual explanation.

Label flip rate. Table 2 provides an overview of the label flip results, which indicates the methods’ ability to find a counterfactual for a given text document. It is essential to note that, due to the low number of words per dataset (between 11.8 and 20.8), it becomes more challenging for methods to find a counterfactual. Indeed, there are fewer candidate words to modify. We observe that, except for the spam detection dataset, transparent methods achieve the highest label flip rates. This underscores the effectiveness of word replacement with antonyms as a means of discovering counterfactuals. Additionally, xSPELLS demonstrates strong performance for the spam detection dataset and exhibits label flip rates similar to transparent methods for polarity detection.

It is crucial to highlight that the spam detection dataset presents increased difficulty due to the presence of numerous special characters and the informal nature of SMS language. This complexity makes generating counterfactuals more challenging. Furthermore, we emphasize that both Growing Net and Growing Language can be fine-tuned for a more exhaustive search by adjusting their parameters, such as reducing the minimal similarity threshold (θ_{min} in Algorithm 3) or exploring more of WordNet’s tree structure (increasing t in Algorithm 2). While such adjustments may enhance the label flip rate, they may also result in longer runtimes.

Runtime. Finally, our results regarding execution time are presented in Table 3. The table details the mean and standard deviation of execution time for each counterfactual explanation method across datasets and classifiers. Notably, cfGAN and Growing Net emerged as the fastest methods for generating counterfactuals. However, it is important to note that cfGAN requires training the Generative Adversarial Network (GAN) on each specific dataset, a process that entails significant training time. The time required for fine-tuning varies, ranging from 4300 seconds for fake news title detection to 6755 seconds for spam detection.

Additionally, we observe that xSPELLS and Growing Language exhibit the slowest performance in terms of execution time. Growing Language, for instance, requires approximately 60 seconds to generate a single counterfactual, while xSPELLS displays execution times ranging from 16 seconds for fake news detection to 219 seconds for spam detection. These findings reveal that, unlike opaque methods like xSPELLS, transparent approaches such as Growing Net are sufficiently fast for real-time explainability.

dataset	method	MLP	RF	BERT
fake	SEDC	31 (14)	13 (6)	15 (3)
	Grow. Net	2 (1)	1 (1)	7 (1)
	Grow. Lang.	55 (28)	55 (13)	34 (12)
	Polyjuice	38 (8)	70 (185)	29 (4)
	cfGAN	1 (0)	1 (0)	1 (0)
	xSPELLS	84 (6)	86 (7)	16 (1)
spam	SEDC	21 (13)	16 (9)	16 (6)
	Grow. Net	1 (1)	1 (1)	11 (4)
	Grow. Lang.	60 (16)	57 (14)	88 (43)
	Polyjuice	32 (7)	62 (184)	33 (15)
	cfGAN	1 (0)	1 (0)	1 (0)
	xSPELLS	219 (17)	198 (16)	22 (1)
polarity	SEDC	13 (10)	12 (9)	21 (6)
	Grow. Net	1 (1)	1 (1)	9 (2)
	Grow. Lang.	75 (33)	74 (32)	65 (29)
	Polyjuice	81 (30)	82 (48)	29 (4)
	cfGAN	1 (0)	1 (0)	1 (0)
	xSPELLS	136 (19)	115 (11)	24 (2)

Table 3: Average runtime in seconds of the studied counterfactual methods (and standard deviation).

8 Discussion & Conclusion

Our evaluation provides valuable insights into the landscape of counterfactual explanations for downstream NLP tasks. One of the most striking findings is that complexity, often associated with the use of neural networks and latent spaces, does not necessarily equate to superior performance in this context. Surprisingly, our results demonstrate that simpler approaches, characterized by a systematic and judicious strategy for word replacement, may yield satisfactory outcomes across different quality dimensions. The results of our study prompt a deeper reflection on the optimal strategies for generating counterfactual explanations in the field of NLP. It invites readers to embrace simplicity and transparency whenever the constraints of the application allow it.

Furthermore, our findings underscore the critical importance of transparency and interpretability in AI and ML, especially in high-stakes applications. The paradox of explaining a black box with another one calls into question the development of opaque approaches when transparent methods suffice, or when transparency is one of the goals in the first place. When focused on NLP applications, our results also call for reflection on the meaning and goal of explanations. If the task is to understand which aspects of a text should change to get a different outcome, a counterfactual explanation that drastically changes every word in the text may not be understandable. On the contrary, a counterfactual based

on simple word-masking, albeit simple, may be perceived as implausible. This could hamper the goal of explanations as a means to elicit trust in users.

That said, the results of our study should be put in context, as the evaluation was conducted on three well-studied downstream applications, namely polarity analysis, fake news detection, and spam detection. Our results might therefore not generalize to other NLP tasks in specialized domains or different languages. While this work puts transparent approaches in the spotlight, our results suggest that plausible counterfactual examples need external domain-adapted knowledge either in the form of language models or knowledge graphs. These may not always be available though. Also, our evaluation was based on popular criteria and metrics for counterfactual explanations. Specialized applications may still take into account additional criteria such as diversity or actionability.

Through this work, we expect to encourage the development of more transparent and interpretable AI systems that foster trust and accountability in every step of the AI-driven decision-making processes, either for prediction, recommendation, or explanation.

Bibliography

- [1] Bodria, F., Giannotti, F., Guidotti, R., Naretto, F., Pedreschi, D., Rinzivillo, S.: Benchmarking and survey of explanation methods for black box models. *Proc. Data Mining and Knowledge Discovery* (2023). <https://doi.org/10.1007/S10618-023-00933-9>, <https://doi.org/10.1007/s10618-023-00933-9>
- [2] Brown, T.B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D.M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D.: Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) *Proc. NeurIPS* (2020), <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>
- [3] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) *Proc. NAACL-HLT. Association for Computational Linguistics* (2019). <https://doi.org/10.18653/v1/n19-1423>, <https://doi.org/10.18653/v1/n19-1423>
- [4] Fellbaum, C.: *WordNet: An Electronic Lexical Database*. Bradford Books (1998)
- [5] Gómez Hidalgo, J.M., Bringas, G.C., Sáenz, E.P., García, F.C.: Content based sms spam filtering. In: *Proc. Symposium on Document Engineering. Association for Computing Machinery, New York, NY, USA* (2006). <https://doi.org/10.1145/1166160.1166191>, <https://doi.org/10.1145/1166160.1166191>
- [6] Guidotti, R.: Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery* (Apr 2022), <https://doi.org/10.1007/s10618-022-00831-6>
- [7] Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S.R., Smith, N.A.: Annotation artifacts in natural language inference data. In: Walker, M.A., Ji, H., Stent, A. (eds.) *Proc. NAACL-HLT. Association for Computational Linguistics* (2018). <https://doi.org/10.18653/v1/n18-2017>, <https://doi.org/10.18653/v1/n18-2017>
- [8] Hase, P., Bansal, M.: Evaluating explainable AI: which algorithmic explanations help users predict model behavior? In: Jurafsky, D., Chai, J., Schluter, N., Tetreault, J.R. (eds.) *Proc. ACL. Association for Computational Linguistics* (2020). <https://doi.org/10.18653/v1/2020.acl-main.491>, <https://doi.org/10.18653/v1/2020.acl-main.491>
- [9] Honnibal, M., Montani, I.: spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing (2017), to appear

- [10] Jacovi, A.: Trends in explainable AI (XAI) literature. CoRR **abs/2301.05433** (2023). <https://doi.org/10.48550/arXiv.2301.05433>, <https://doi.org/10.48550/arXiv.2301.05433>
- [11] Li, C., Shengshuo, L., Liu, Z., Wu, X., Zhou, X., Steinert-Threlkeld, S.: Linguistically-informed transformations (LIT): A method for automatically generating contrast sets. In: Alishahi, A., Belinkov, Y., Chrupala, G., Hupkes, D., Pinter, Y., Sajjad, H. (eds.) Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2020, Online, November 2020. pp. 126–135. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.blackboxnlp-1.12>, <https://doi.org/10.18653/v1/2020.blackboxnlp-1.12>
- [12] Li, Z., Tao, R., Wang, J., Li, F., Niu, H., Yue, M., Li, B.: Interpreting the latent space of gans via measuring decoupling. IEEE Trans. Artif. Intell. (2021), <https://doi.org/10.1109/TAI.2021.3071642>
- [13] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR (2019), <http://arxiv.org/abs/1907.11692>
- [14] Madaan, N., Padhi, I., Panwar, N., Saha, D.: Generate your counterfactuals: Towards controlled counterfactual generation for text. In: Thirty-Fifth Conference on Artificial Intelligence, AAAI, Conference on Innovative Applications of Artificial Intelligence, IAAI, The Symposium on Educational Advances in Artificial Intelligence, EAAI. AAAI Press (2021), <https://ojs.aaai.org/index.php/AAAI/article/view/17594>
- [15] Martens, D., Provost, F.J.: Explaining data-driven document classifications. MIS Q. (2014), <http://misq.org/explaining-data-driven-document-classifications.html>
- [16] McCoy, T., Pavlick, E., Linzen, T.: Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In: Korhonen, A., Traum, D.R., Marquez, L. (eds.) Proc. ACL. Association for Computational Linguistics (2019). <https://doi.org/10.18653/v1/p19-1334>, <https://doi.org/10.18653/v1/p19-1334>
- [17] Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artif. Intell. (2019), <https://doi.org/10.1016/j.artint.2018.07.007>
- [18] Morris, J.X., Lifland, E., Yoo, J.Y., Grigsby, J., Jin, D., Qi, Y.: Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in NLP. In: Liu, Q., Schlangen, D. (eds.) Proc. EMNLP. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.emnlp-demos.16>, <https://doi.org/10.18653/v1/2020.emnlp-demos.16>
- [19] Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proc. ACL (2005)
- [20] Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proc. EMNLP. Association for Computational Linguistics (2019), <http://arxiv.org/abs/1908.10084>

- [21] Robeer, M., Bex, F., Feelders, A.: Generating realistic natural language counterfactuals. In: Findings EMNLP. Association for Computational Linguistics (2021), <https://doi.org/10.18653/v1/2021.findings-emnlp.306>
- [22] Ross, A., Marasovic, A., Peters, M.E.: Explaining NLP models via minimal contrastive editing (mice). In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Findings ACL/IJCNLP. Association for Computational Linguistics (2021). <https://doi.org/10.18653/v1/2021.findings-acl.336>, <https://doi.org/10.18653/v1/2021.findings-acl.336>
- [23] Ross, A., Wu, T., Peng, H., Peters, M.E., Gardner, M.: Tailor: Generating and perturbing text with semantic controls. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proc. ACL. Association for Computational Linguistics (2022). <https://doi.org/10.18653/v1/2022.acl-long.228>, <https://doi.org/10.18653/v1/2022.acl-long.228>
- [24] S. Punla, C., C. Farro, R.: Are we there yet?: An analysis of the competencies of BEED graduates of BPSU-DC. *International Multidisciplinary Research Journal* **4**(3), 50–59 (Sep 2022)
- [25] Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. ArXiv **abs/1910.01108** (2019)
- [26] Shen, Y., Gu, J., Tang, X., Zhou, B.: Interpreting the latent space of gans for semantic face editing. In: Proc. CVPR. Computer Vision Foundation / IEEE (2020). <https://doi.org/10.1109/CVPR42600.2020.00926>, https://openaccess.thecvf.com/content_CVPR_2020/html/Shen_Interpreting_the_Latent_Space_of_GANs_for_Semantic_Face_Editing_CVPR_2020_paper.html
- [27] Verma, S., Dickerson, J.P., Hines, K.: Counterfactual explanations for machine learning: A review. In: NeurIPS 2020 Workshop: ML Retrospectives, Surveys & Meta-Analyses ML-RSA. vol. abs/2010.10596 (2020). <https://doi.org/https://arxiv.org/abs/2010.10596>
- [28] Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law and Technology* **31**(2), 841–87 (2018)
- [29] Wei, X., Ngo, C.: Ontology-enriched semantic space for video search. In: Proc. International Conference on Multimedia. ACM (2007), <https://doi.org/10.1145/1291233.1291447>
- [30] Wu, T., Ribeiro, M.T., Heer, J., Weld, D.S.: Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proc. ACL/IJCNLP. Association for Computational Linguistics (2021). <https://doi.org/10.18653/v1/2021.acl-long.523>, <https://doi.org/10.18653/v1/2021.acl-long.523>
- [31] Yang, L., Kenny, E.M., Ng, T.L.J., Yang, Y., Smyth, B., Dong, R.: Generating plausible counterfactual explanations for deep transformers in financial text classification. In: Proc. COLING. International Committee on Computational Linguistics (2020). <https://doi.org/https://doi.org/10.18653/v1/2020.coling-main.541>

A Counterfactual Generation

We begin by describing the six counterfactual generation methods used to generate counterfactuals. We fill the middle ground with two methods, Growing Net and Growing Language, which implement a strategy similar to existing transparent methods. However, they do so with fewer methodological complexities. We adapted the code used to generate counterfactuals for the three transparent methods (SEDC, Growing Net, and Growing Language) and made it available on GitHub⁷. Conversely, we used the original code for the opaque methods, as described below.

SEDC: We modified the code used for word masking to ensure its compatibility with classification models that do not output class probabilities. This modified code version is accessible in Python on our GitHub as a variant of the counterfactual method class. This class proposes to choose from SEDC, Growing Net, or Growing Language, all specialized in generating transparent explanations.

Polyjuice: To generate counterfactuals, we used the code available at the official link <https://github.com/tongshuangwu/polyjuice>. We used default hyperparameters with the use of all control codes to perturb texts from each test set until we found 100 instances classified differently by the model.

xSPELLS: We used the V2 version of xSPELLS, available on GitHub <https://github.com/lstate/X-SPELLS-V2>, with default hyperparameters.

CounterfactualGAN: We used the code provided in the official release page of the paper, which can be accessed at <https://aclanthology.org/2021.findings-emnlp.306/>. We executed CounterfactualGAN (cfGAN) with default hyperparameters.

This comprehensive approach to counterfactual generation ensures a diverse set of methods to be evaluated and compared in our experiments.

B Illustrative Example

We provide in this section, some examples of counterfactuals generated for each method and each dataset.

B.1 Fake News Detection

Original Text: Obama To Apply For Political Asylum In Moneygall
 SEDC: **MASK MASK MASK** For Political Asylum In Moneygall
 Growing Net : Obama To **hold** For Political Asylum In Moneygall
 Growing Language : Obama To Apply For **Hilarious** Asylum In Moneygall
 Polyjuice : Obama is **expected** To apply for political asylum in **Guantanamo Bay**
 cfGAN : N/A
 xSPELLS : **why most states are struggling to**

⁷ <https://github.com/j2launay/ebbwb>

B.2 Spam Detection in SMS

Original Text: Sunshine Quiz Wkly Q! Win a top Sony DVD player if u know which country the Algarve is in? Txt ansr to 82277. aPS1.50 SP:Tyrone

SEDC: **MASK MASK** Wkly Q ! Win **MASK** top **MASK MASK MASK** if u know **MASK MASK** the Algarve is in ? **MASK** ansr **MASK. MASK** Tyrone

Growing Net: **sun test** Wkly Q ! **bring home** a **clear** Sony **videodisc musician** if u know which country the Algarve is in ? Txt ansr to 82277 . aPS1.50 SP : Tyrone

Growing Language: **Europe John** Wkly Q ! **Rest** a **technical Dr Laptop** player if **sis** know which country the Algarve **feels** in ? Txt ansr to 82277 . aPS1.50 **Gen. – Michigan**

Polyjuice: **shine** quiz wkly q! win **wkly**

cfGAN: **##cher week wk mobile two!** win a **as earthhayaphonic** if u know which country the **chance week o** is in? **tt opposed and send fin** gives

xSPELLS: **you were your each re not supposed and collect is good way u any please send 50 reply after**

B.3 Polarity Detection

Original Text: This is one of Polanski's best films

SEDC: This is one of **MASK MASK MASK**

Growing Net: This is one of Polanski's **ill** films

Growing Language: This is one of Polanski's **worst** films

Polyjuice: This is one of Polanski's **worst movies**

cfGAN: This is one of **shot kingdom intelligence's all**

xSPELLS: **N/A**