Relevance-based Infilling for Natural Language Counterfactuals

Lorenzo Betti
ISI Foundation
Turin, Italy
Department of Network and Data Science, Central European University
Vienna, Austria
lrn.betti@gmail.com

Carlo Abrate
CENTAI
Turin, Italy
Sapienza University
Rome, Italy
carlo.abrate@centai.eu

Francesco Bonchi
CENTAI
Turin, Italy
Eurecat
Barcelona, Spain
bonchi@centai.eu

Andreas Kaltenbrunner
ISI Foundation
Turin, Italy
Universitat Oberta de Catalunya
Barcelona, Spain
kaltenbrunner@gmail.com

ABSTRACT

Counterfactual explanations are a natural way for humans to gain understanding and trust in the outcomes of complex machine learning algorithms. In the context of natural language processing, generating counterfactuals is particularly challenging as it requires the generated text to be fluent, grammatically correct, and meaningful. In this study, we improve the current state of the art for the generation of such counterfactual explanations for text classifiers. Our approach, named RELITC (Relevance-based Infilling for Textual Counterfactuals), builds on the idea of masking a fraction of text tokens based on their importance in a given prediction task and employs a novel strategy, based on the entropy of their associated probability distributions, to determine the infilling order of these tokens. Our method uses less time than competing methods to generate counterfactuals that require less changes, are closer to the original text and preserve its content better, while being competitive in terms of fluency. We demonstrate the effectiveness of the method on four different datasets and show the quality of its outcomes in a comparison with human generated counterfactuals.[1]

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**.

KEYWORDS

NLP; masked language model; explainability; counterfactuals

ACM Reference Format:
Lorenzo Betti, Carlo Abrate, Francesco Bonchi, and Andreas Kaltenbrunner. 2023. Relevance-based Infilling for Natural Language Counterfactuals. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3583780.3615029

[1]The code used to perform the experiments is available on GitHub at https://github.com/Loreb92/relitc-counterfactuals.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3615029

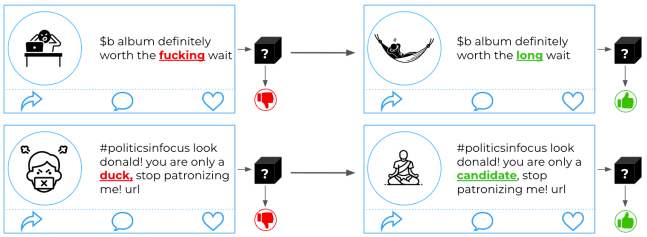

Figure 1: Two examples of counterfactuals from the OLID dataset [40]. Given a tweet (on the left) classified as offensive by the black-box classifier, RELITC generates a tweet that is as close as possible to the original one and flips the prediction.

1 INTRODUCTION

The recent breakthroughs in the ability to train large language models (LLMs) have pushed NLP techniques to achieve impressive performances on a wide variety of tasks and to be used in high-stake decisions and applications across widespread types of industries. LLMs are composed of millions or billions of parameters and trained over large corpora, so that only few companies are able to produce them. This rises several ethical concerns. Firstly, these models are generally considered opaque *black-box* models [24] due to their complexity. Secondly, the proprietary nature of some LLMs make them inaccessible to users and developers, e.g., because of commercial secrecy and intellectual property [9]. These issues can be obstacles towards the responsible deployment of these models and their indiscriminate use can exacerbate or even create new types of biases [29]. Consequently, developing explainability methods for NLP is a goal of uttermost importance.

Given the complexity of LLMs, developing methods that produce interpretable models based on a limited number of human-interpretable features seems impractical. For this reason, the bulk of the effort has been so far devoted to techniques that produce *post-hoc* explanations [6]. An important group of post-hoc methods is based on *counterfactuals*. Supported by cognitive science, the idea at the basis of counterfactuals is that of explaining the reason of an event comparing it with an event that did not occur [26], while most of the potential causes leading to it, still were present. In other terms, the goal of counterfactual explanations is to show what a machine learning model considers a valid opposite example [24]. In the context of an NLP classification task, a counterfactual is a piece of text that, while being as similar as possible to the original text, is classified differently by the model.

More formally, given a text $x \in \mathcal{X}$ composed of a set of N tokens $x = [t_1, t_2, ..., t_N]$, and a binary black-box classifier $\mathbf{B} : \mathcal{X} \rightarrow [0, 1]$ whose prediction on x is $\mathbf{B}(x) \in [0, 1]$, a text $\tilde{x}$ is a counterfactual of x if the following conditions are fulfilled:

(1) *counterfactual class*: $\tilde{x}$ is in a different class, $\mathbf{B}(\tilde{x}) \neq \mathbf{B}(x)$;
(2) *closeness*: x and $\tilde{x}$ differ only by minimal lexical changes;
(3) *feasibility and content preservation*: $\tilde{x}$ is a feasible text ($\tilde{x} \in \mathcal{X}$) and the content of x is preserved.

We denote with $\mathbf{B}(x) = y$ the original label of x and $\mathbf{B}(\tilde{x}) = y_c$ the counterfactual label. Two examples are shown in Figure 1.

Defining accurate and reliable counterfactual methods for NLP tasks is an open research question with applications going beyond explainability. One such example is the assessment of the fairness of black-box classifiers with respect to sensitive attributes. Indeed, divergent predictions by classifiers for texts varying solely in the mentioned identity groups suggest an unfair behavior of the model and necessitate analyst intervention. Counterfactuals can aid in creating metrics to assess how predictions change when sensitive attributes are changed [14]. Counterfactuals have a potential application in text detoxification as well. Given a sentence classified as inadequate, its counterfactual would be an alternative text that retains the original content with minimal lexical changes, but is no longer labeled as toxic by the classifier. This allows the author of the text to discern triggering elements and receive suggestions for alternative phrasing. Therefore, textual counterfactuals offer a valuable and actionable response to contrast hate-speech or discriminatory attitudes online. Other applications of counterfactuals include model auditing and debugging [19], identifying spurious correlations in datasets [46], model debiasing [34], and the assessment of factual consistency in automatic text summarization [51].

Challenges and contributions. The main challenge in generating textual conterfactuals is producing a text that is realistic, in order to be acceptable by humans [26]. This general requirement is faceted, as it is related to plausibility [18], fluency [39], grammatical correctness, and semantic meaningfulness [27]. Furthermore, to generate a good counterfactual we need to respect the "closest possible world" requirement [47], which, in the context of NLP, means that the textual counterfactual must have the minimum amount of edits with respect to the original instance [38, 49].

One approach to address these challenges consists in replacing the words that contribute the most to the prediction of the black box with words that push the prediction to a different label [50]. These words can be first identified and masked through a feature attribution method, and then replaced using a masked language model, i.e. a model trained to infill words that are masked in a text (e.g., BERT [7]). This approach enforces the "closest possible world" requirement by-design, since the original text is modified through word substitutions. However, the order with which these words are infilled may affect the fluency of the generated counterfactuals. As finding the optimal ordering is hard ($n!$ possible orderings for n words to be replaced), recent efforts have proposed pre-training strategies in which words are infilled in random order to make the model perform equally well independently of the chosen ordering. However, the infilling orderings defined for the inference step either employ simple left-to-right [3, 8] or random ordering [21], or use beam-search to approximate the optimal trajectory [42].

In this paper, we introduce RELITC (Relevance-based Infilling for Textual Counterfactuals), a new method for textual counterfactual generation. Given a NLP classifier and an input text, our method first replaces with mask tokens a fraction of the most important words with respect to the label predicted by the classifier. These are identified through a feature importance method. Then, a *conditional masked language model* (CMLM, i.e. a model that can condition its prediction on a desired attribute) outputs a probability distribution over a vocabulary for each mask token, conditioning on the counterfactual label. Mask tokens are infilled by sampling from these probability distributions. To choose the order to infill the mask tokens, we define a novel strategy based on the entropy of their associated probability distributions. Intuitively, the lower the entropy associated to a mask token, the more confident the model in the prediction of the word to replace.

In particular, our main contributions are:

- We introduce RELITC, an approach to generate counterfactuals for text classifiers that first masks important words for the classifier and then infills them through a masked language model conditioned on the counterfactual class. The evaluation of RELITC on four real-world datasets associated to different classification tasks shows that the generated counterfactuals are feasible, close to the input text, effectively flip the predicted label, and preserve well its original content. RELITC generates counterfactuals that are close to human-edited examples as well.
- We define a novel ordering strategy to choose the order with which mask tokens are infilled. This produces counterfactuals that are more fluent compared to the common left-to-right infilling, especially for long texts.
- We show with a small-scale experiment that RELITC achieves comparable performance with both a model-specific (Integrated Gradients [44]) and a model-agnostic feature importance methods (SHAP [22]). This is useful when it is not possible to have full access to the black-box classifier, but it can only be queried.
- We show that RELITC better preserves the content of the original input text requiring less edits than the ones generated by MiCe [39], a state-of-the-art counterfactual text generator, while being competitive in terms of fluency. For one dataset, we conducted a human evaluation that confirms these observations. RELITC is also faster than MiCe in generating counterfactuals.

2 RELATED WORK

NLP explainability. Explainability for NLP [6, 54] is challenging due to the unstructured nature of the input: techniques that are effective with tabular or image data cannot be adapted straightforwardly. Feature importance can be used to generate saliency maps of the most relevant words in the text [28, 48]. Another approach is based on surrogate models, i.e. learning a second, more interpretable model locally [2, 33, 37, 45]. Explainable methods can be applied at different stages [54]: (a) at the embedding phase in the input space [1, 11, 13, 30, 32]; (b) at the model or attention layer where explainability methods attempt to interpret internal states [15, 25]; and (c) at the output layer.

NLP counterfactuals. Textual counterfactual generation shares some aspects with other NLP tasks, namely adversarial attacks and style transfer. The former aims at making imperceptible text

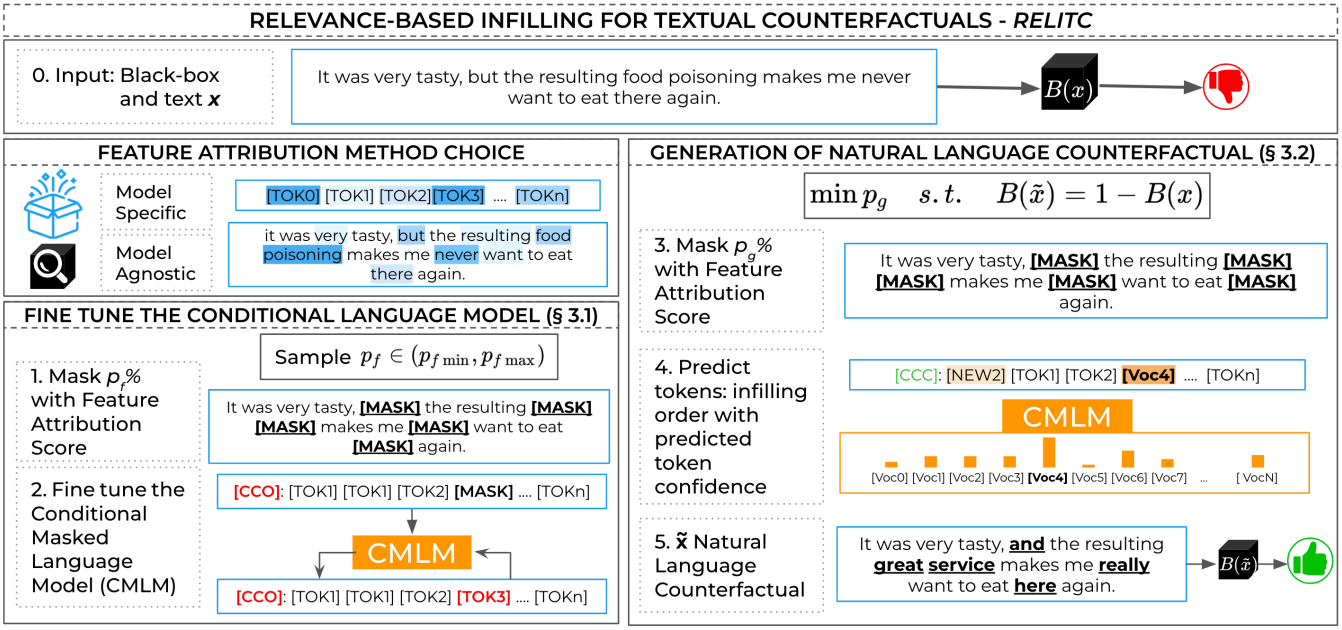

Figure 2: RELITC approach. Given a text and a black-box classifier given as input (step 0), a Feature Attribution Method is selected to return a token-level relevance score. The fine-tuning of the Conditional Masked Language Model (CMLM) (steps 1 and 2) is based on learning to infill a percentage p_f of the most relevant tokens according to the feature attribution method with respect to the original class. The generation procedure (steps 3 to 5) minimizes the fraction p_g of replaced tokens by repeatedly masking the most relevant tokens (step 3), and infilling them according to the confidence of the CMLM (step 4).

perturbations that can cause a model to return incorrect predictions regardless of the feasibility of the output text [53]. The latter aims at changing the style of a text while preserving its content, with no need to be close to the original text in terms of edit distance [17].

Some authors have proposed to generate counterfactuals through text perturbations that are neither task-specific nor black-box dependent. Polyjuice [49] is able to perturb an input text according to 8 different transformations such as paraphrasing, word substitutions, and negations. It is trained on sentence-pairs datasets, one dataset for each transformation. Fryer et al. [12] propose a method based on a language model that can vary sensitive attribute features of the original text using specific prompts, such as race and gender. Although useful for auditing black boxes with respect to general text transformations, these methods are not directly related to the specific task for which the black box was trained.

Other approaches, including the present work, aim at producing task-specific counterfactuals. One possible approach consists in using the predictions of the black box to find perturbations in the latent space [38] or steer the internal states [23] of a language model to generate counterfactuals. Another methodology is "Mask and infill", borrowed from the style transfer literature [50]. It involves two steps: replacing portions of text with blanks and then filling these blanks with texts more relevant to the counterfactual class. In the context of counterfactual generation, Chen et al. [5] propose a model that splits texts into chunks through a syntactic parser and uses GPT3 [4] to infill these spans. The black box is then used to filter counterfactuals that flip the predicted class. However, the black box can inform the selection of text spans to mask. In the case of **MiCe** [39], which is the work most related to our approach, a feature attribution method is employed to identify the most important words for the black box. After the masking step, MiCe uses the T5 model [35] to infill the masked spans conditioning on the counterfactual label, after fine-tuning on a specific dataset. This model is able to make minimal changes to generate fluent counterfactuals thanks to the performance of the T5 language model in text generation. The T5 model of MiCe is fine-tuned to infill masked spans from left to right, thus neglecting other possible trajectories. Our approach aims at addressing these limitations with a model having half of the number of parameters.

3 THE RELITC APPROACH

In this section, we introduce RELITC, our method for generating textual counterfactuals whose core components are summarised in Figure 2. It involves two steps. Firstly, fine-tuning a CMLM to make it able to infill mask tokens conditioning on a desired label (step 1 and 2 in Figure 2). Secondly, the generation of the counterfactual by masking a fraction p_g of its most informative tokens (step 3) and infilling one mask per time using the fine-tuned CMLM conditioned on the counterfactual label. The infilling ordering is guided by the confidence of the model in the prediction of the token (step 4 in Figure 2). We provide details about the fine-tuning of the CMLM in §3.1, while we report the details about the generation phase in §3.2.

3.1 Fine-tuning the conditional language model

RELITC uses a masked language model to infill mask tokens given the context provided by the unmasked words and conditioned on an input label. The fine-tuning of the CMLM is inspired by [39].

Algorithm 1: The infilling mechanism of RELITC.

Input : $masked\_text$, CMLM
Output: $counterfactuals$

Function *Mask_Infill*:
  $loc\_masks \leftarrow$ where_masks($masked\_text$)
  while len(loc_masks) > 0
    $next\_tok\_probs \leftarrow$ CMLM($masked\_text$)
    $tok2infill \leftarrow$ select_token($loc\_masks, next\_tok\_probs$)
    $masked\_text \leftarrow$ infill_token($next\_tok\_probs, tok2infill$)
    $loc\_masks \leftarrow$ where_masks($masked\_text$)
  end
  $counterfactuals \leftarrow masked\_text$
  return $counterfactuals$
end

First, a feature attribution method assigns a relevance score to all the words in the input text, which quantifies the contribution of each word in the prediction of the black box.[2] Then, a fraction p_f of the most important tokens is replaced with a mask token, with p_f sampled uniformly in the interval $(p_{f,\min}, p_{f,\max})$ (step 1 in Figure 2).

Afterwards, a control code corresponding to the label predicted by the black box is prepended to the instace (step 2). For example, in a sentiment classification task, the input sequence is prepended with the control code *"positive:"* (*"negative:"*) to condition on the positive (negative) class. Intuitively, the control code makes the CMLM learn correlations between classification labels and words related to a specific class.

Finally, the CMLM is fine-tuned to predict the words that were masked using the masked language modeling loss [7] (step 2). This allows tailoring the model to the specific task by letting the CMLM focus on the most relevant words for the black box.

We employ a pre-trained BERT model as the conditional masked language model of RELITC and Integrated Gradients as the feature attribution method in our experiments, which requires to have access to the black-box model (*model-specific*). In case this is not possible since the black-box classifier can only be queried, it could be replaced by a *model-agnostic* method.

3.2 Textual counterfactuals generation

The CMLM can then be used to infill mask tokens conditioned on the counterfactual label. Given an input text, we mask the input with a procedure similar to the one described in the fine-tuning step (not necessarily using the same feature attribution method). In particular, we mask a fraction p_g of the most important words of the text (step 3 in Figure 2) and then prepend the control code corresponding to the counterfactual label instead of the predicted label (step 4). This allows the fine-tuned model to infill mask tokens conditioning on the counterfactual label. The infilling process is illustrated in Algorithm 1. The masked input text is fed into the model to output one probability distribution across the model's vocabulary for each mask token (line 4). Instead of replacing all masked tokens in a single forward pass, we select one mask token to be replaced (lines 5 and 6) and fed the resulting text again into the model (step 4 and lines 3-8). The benefit consists in the model having more context to predict the next token after each step.

[2]Whenever the tokenizer splits a word into subwords, their relevance scores are aggregated through max pooling to prevent uninterpretable word splits.

Algorithm 2: RELITC Search for the minimal mask fraction

Input : $text$, BlackBox, CMLM, $token\_scores$, y_c, $p_{g,\min}, p_{g,\max}$, m
Output: $counterfactuals, predictions$
Global : $counterfactuals \leftarrow$ empty list
  $predictions \leftarrow$ empty list

1 **while** $m > 0$ & $(1/p_{g,\min} - 1/p_{g,\max} > 1/\#\text{tokens}(text))$
2 $p_g \leftarrow (p_{g,\min} + p_{g,\max})/2$
3 $maskedText \leftarrow$ mask_text($text, p_g, token\_scores$)
4 $maskedText \leftarrow$ add_ctrl_code($masked\_text, y_c$)
   /* call Alg. 1 to infill *maskedText* w. CMLM */
5 $newCnterfactls \leftarrow$ Mask_Infill($maskedText$, CMLM)
6 $new\_preds \leftarrow$ BlackBox($newCnterfactls$)
7 $counterfactuals$.append($newCnterfactls$)
8 $predictions$.append(new_preds)
9 **if** any$(new_preds == y_c)$
10 $p_{g,\max} \leftarrow p_g$
11 **else**
12 $p_{g,\min} \leftarrow p_g$
13 **end**
14 $m \leftarrow m - 1$
15 **end**

However, there is no a priori order to infill the mask tokens. Usually, infilling proceeds from left to right, but there is no reason to consider this ordering optimal. Therefore, after having tested multiple infilling strategies, we define an infilling order informed by the model's confidence in the prediction of the next token. Specifically, we compute the entropy associated with the output probability distributions and choose to infill the mask token having the lowest entropy. Intuitively, the lower the entropy, the more peaked is the probability distribution around a few tokens, thus signaling the model being more confident about the prediction. This strategy has a twofold benefit: first, at each step, we infill the mask token for which the model is more confident, and second, mask tokens for which the model is uncertain are infilled later, thus taking advantage of a larger available context.

Finally, the optimal mask fraction p_g is determined through a binary search in the interval $(p_{g,\min}, p_{g,\max})$, whose aim is to minimize the mask fraction needed to generate a counterfactual [39]. In that way, only the minimum number of tokens is changed to flip the prediction of the black box. This process is shown in Algorithm 2. The binary search starts by masking a fraction p_g of the most relevant words of the input text (lines 2-3), according to the feature attribution method. Then, the control code of the counterfactual label is prepended to the masked text and the CMLM generates N candidate counterfactual examples from the masked text (lines 4-5). The black box predicts the labels of the candidate counterfactuals

(line 6) and, if at least one candidate counterfactual is classified in the counterfactual class y_c, we consider it a successful generation, otherwise the generation is a failure (line 9). In the former case, we replace $p_{g,\max}$ with p_g, thus decreasing the mask fraction of the next step of the binary search (line 10). In case of failure, the mask fraction is increased (lines 12). We stop the binary search at the m-th level (line 1) or when the updated values of $p_{g,\min}$ and $p_{g,\max}$ would both lead to the same number of masked tokens used in the previous level of the search.

We set the hyper-parameters similarly to [39], with $p_{f,\min} = 0.20$, $p_{f,\max} = 0.55$, $p_{g,\min} = 0.0$, $p_{g,\max} = 0.50$, and a maximum level $m = 4$ for the binary search. We sample $N = 15$ candidate counterfactuals during each round using nucleus sampling [10, 16] for generation with $top_k = 50$ and $top_p = 0.95$.

3.3 Evaluation metrics

We employ different metrics to automatically evaluate the quality of the counterfactuals of RELITC and different baselines, comparing them to the original input text and human-edited counterfactuals.

Flip rate quantifies for how many instances the tested approach is able to generate a counterfactual, i.e. a text whose predicted label corresponds to the counterfactual label. It is defined as the fraction of successful counterfactual examples. (the higher, the better)

Normalized Edit Distance (NED) [39] quantifies the closeness between the original and the counterfactual texts. It is defined as the Levenshtein distance between the original text x and the generated counterfactual $\tilde{x}$ divided by the number of words in x. The Levenshtein distance corresponds to the minimum number of substitutions, insertions, or deletions of words required to change one text into the other. It ranges from 0 to 1, where 0 indicates that the two texts are identical. (the lower, the better)

Content preservation quantifies to what extent the content of the input text is preserved in a generated counterfactual. To this end, we use SBERT [36] to encode both the original input and the counterfactual text into two vectors, $\mathbf{x}$ and $\tilde{\mathbf{x}}$ respectively. Then, the content preservation is defined as the cosine similarity between the two vectors. We used the *all-mpnet-base-v2* model[3]. The measure ranges from -1 to 1, where 1 indicates that the two texts are maximally similar. (the higher, the better)

Fluency quantifies how well a counterfactual text reads in English in comparison to the input text and can be considered as a proxy for plausibility [39]. Fluency is computed automatically using a masked language model. Given a text composed of a sequence of N tokens $x = [t_1, t_2, ..., t_N]$, let $x_{\setminus t_i} = [t_1, ..., t_{i-1}, MASK, t_{i+1}, ..., t_N]$ be the same sequence as x in which the token t_i is replaced with a mask token. Following Ref. [40], we define the normalized masked language model score of x as:

$$MLM\ score\ (x) = \frac{1}{N}\sum_{i=1}^{N} \log P_{MLM}\left(t_i | x_{\setminus t}; \Theta\right)$$

where θ denote the parameters of the masked language model and $P_{MLM}\left(t_i | x_{\setminus t_i}; \Theta\right)$ is the probability returned by the masked language model that the token t_i can replace the mask token in $x_{\setminus t_i}$. Intuitively, this score quantifies how likely the sequence x is for the masked language model and has been shown to perform well in linguistic acceptability tasks [40]. The fluency of the generated counterfactual example $\tilde{x}$ against x is then defined as:

$$fluency\ (\tilde{x}; x) = \frac{MLM\ score\ (\tilde{x})}{MLM\ score\ (x)}.$$

When close to 1, this score indicates that the the sequences x and $\tilde{x}$ are equally likely for the masked language model. On the other hand, when larger (lower) than 1, the counterfactual example is less (more) likely than the original input text. We used the *bert-base-uncased* pre-trained model as the masked language model[4]. (the closer to 1, the better)

Mask Fraction is the fraction of masked tokens p_g defined in §3.2. It corresponds to the minimum fraction of tokens needed to generate a successful counterfactual. (the lower, the better)

BLEU was originally introduced to evaluate the accuracy of machine translations with respect to a reference translation [31]. BLEU can also be used to compare generated counterfactuals with reference texts edited by humans. It takes values between 0 and 1, and the higher the BLEU the closer the generated text to the human reference. (the higher, the better)

4 EVALUATION

In this section we present the experiments aimed at assessing the effectiveness of our approach. In all the experiments we generate counterfactuals for a machine-learning classifier trained on a specific task in a specific dataset and use the very same classifier to decide whether the output texts of our algorithm are indeed counterfactuals (i.e. they fall into a different class). All experiments are performed on a NVIDIA Quadro RTX 6000 (24GB).

We compare our approach against few non-trivial baselines, a competing method (§4.3), and counterfactuals generated manually by human annotators (§4.4). Before presenting the results, we briefly describe the datasets (§4.1) and the baseline methods (§4.2).

4.1 Datasets

We perform experiments on the following datasets:

Yelp[5] is a sentiment classification task already split into 560,000 train and 38,000 test instances, balanced between two classes. Texts correspond to reviews of business activities and labels to their sentiment. The black box we use is trained on this training set[6]. For our experiments we sample 100,000 train instances for training and 10,000 test instances to generate counterfactuals. We use 20% of the training instances as validation set. After truncating all reviews to 256 tokens (due to limits in our GPU capacity), there are 111 words per instance on average.

OLID [52] is a two-class offensiveness detection task containing 13,240 train (33% offensive) and 860 test (28% offensive) tweets annotated using crowd-sourcing. The black box used is trained on this training set[7]. We train the counterfactual methods on the same dataset and generate counterfactuals on the test set. We use 20% of the training instance as validation set. After removing mentions and urls, there are 23 words per tweet on average.

[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

[4] https://huggingface.co/bert-base-uncased

[5] https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset

[6] https://huggingface.co/textattack/bert-base-uncased-yelp-polarity

[7] https://huggingface.co/cardiffnlp/twitter-roberta-base-offensive

Table 1: Results of RELITC on the Yelp and OLID datasets, using the left-to-right (L2R) and confidence (Conf.) infilling orderings. Values refer to the average of the corresponding metric across successful counterfactuals. Bold texts refer to the best score for each metric, and symbols indicate that RELTIC is significantly better than MiCe (Welch's t-test statistic: * $p < 0.05$; ** $p < 0.001$).

	YELP					OLID				
	Flip rate ↑	NED ↓	Fluency (≈ 1)	Content ↑ preserv.	Mask ↓ Frac.	Flip rate ↑	NED ↓	Fluency (≈ 1)	Content ↑ preserv.	Mask ↓ Frac.
BERT_{L2R}	0.742	0.211	1.021	0.813	0.240	0.579	0.307	0.910	0.789	0.195
$\text{BERT}_{\text{Conf}}$	0.743	0.208	**0.991**	0.813	0.239	0.614	0.334	0.887	0.780	0.208
MiCe	**0.999**	0.239	1.041	0.837	0.175	**1.000**	0.246	**1.037**	0.749	0.187
$\text{RELITC}_{\text{L2R}}$	0.948	0.129**	1.075	0.900**	**0.170***	0.999	**0.199****	1.059	**0.882****	**0.086****
$\text{RELITC}_{\text{Conf}}$	0.943	**0.124****	1.046	**0.901****	0.171*	0.998	0.204**	1.046	0.877**	0.094**

Yelp sentence [20, 43] is a sentence-level sentiment classification task. It contains 443,259 train, 4000 validation, and 1000 test instances. The latter two splits are balanced, while 60% of the sentences in the training set are positive. We fine-tune the black box starting from a *bert-base-uncased* pre-trained model and fine-tune the counterfactual models on the same training set. Then, the test set is used for the generation of counterfactuals. In addition, for each instance in the test set there is one human-edited counterfactual as reference. There are 8 words per sentence on average.

Call me sexist but... [41] contains tweets labeled as sexist and non-sexist annotated using crowd-sourcing. We will refer to this dataset as "CallMe". We split the dataset into 6982 train, 1745 validation, and 3728 test examples, all splits containing 85% of non-sexist tweets. We fine-tune the black box starting from a *bert-base-uncased* pre-trained model and fine-tuned the counterfactual models on the same training set. Tweets in the test set were used to generate counterfactuals. The dataset contains human-edited examples of tweets obtained through crowd-sourcing, whose original class is sexist. There are 383 of such examples in the test set. The tweets contain 15 words on average.

4.2 Baselines

The first baseline consists in using the left-to-right (L2R) ordering to infill the masked words instead of our proposed ordering, which is informed by the confidence of the masked language model. This provides a comparison between the two infilling orderings. We name these two models $\textbf{RELITC}_{\text{L2R}}$ and $\textbf{RELITC}_{\text{Conf}}$ respectively.

In addition, we compare the results of $\text{RELITC}_{\text{Conf}}$ against two baselines and one-state-of-the-art method:

BERT: these baselines are based on the pre-trained *bert-base-uncased*. The only difference with RELITC is that the baselines are not fine-tuned on a specific dataset and can only rely on the control code to condition on the counterfactual label. Thus, these baselines inform about how RELITC benefits from the task-specific fine-tuning in conditioning the text infilling towards the desired counterfactual class. We distinguish between $\textbf{BERT}_{\text{L2R}}$ and $\textbf{BERT}_{\text{Conf}}$ depending on the ordering strategy used.

MiCe [39]: we compare our approach with MiCe, a state-of-the-art model that shares a similar framework and addresses the same problem as RELITC. MiCe aims to generate minimal and fluent counterfactuals for NLP tasks, which aligns with the objective of our proposed methodology. Details about MiCe and the main differences with RELITC are discussed in §2.

4.3 Comparison with other methods

In this section we compare MiCe [39] and the BERT baselines mentioned above with our RELITC approach.

Comparisons based on averages over a dataset: We start with the Yelp and Olid datasets. The corresponding results are shown in Table 1. First, we compare the **BERT** baselines to our **RELITC**, regardless of the infilling strategy. As expected, the BERT baselines lead to substantially lower flip rates than our RELITC model (less than 75% for YELP and 62% for OLID vs. above 94% and 99% respectively). This shows that fine-tuning the model is effective in generating counterfactuals that flip the label predicted by the classifier. At the same time, fewer tokens need to be changed (lower mask fraction and lower NED) and the content is closer to the input text (Content preserv. column)[8].

When comparing the two ordering strategies of RELTIC we observe that **Confident** generates counterfactuals whose fluency is lower than the ones generated through **L2R**. This can be best observed in the Yelp dataset where the fluency has a relative improvement of 3% (from 1.075 for L2R to 1.046 for Confident), and might be explained by the larger input text length of YELP. Indeed, the differences between the two orderings can be better appreciated in longer texts because, for a fixed mask fraction, more words are masked and the infilling trajectories may show larger differences. In preliminary experiments on the Yelp dataset (data not shown), we have also evaluated RELITC with right-to-left and inverse confidence orderings (i.e. mask tokens are infilled from least to most confident). We observed right-to-left to perform similar as left-to-right, while the fluency of the inverse confidence ordering degrades compared to $\text{RELITC}_{\text{Conf}}$.

The comparison between $\textbf{RELITC}_{\text{Conf}}$ and **MiCe** shows that our approach is able to generate counterfactual examples that better preserve the content of the original input text and are closer to the original text (lower NED: averages of 0.124 vs 0.239 for YELP and 0.204 vs. 0.246 for OLID), while being comparable in terms of fluency (averages of 1.046 vs 1.041 for YELP and 1.046 vs. 1.037 for OLID). All those differences are statistically significant. In the OLID dataset we also need to change a significantly lower fraction of tokens (mask fraction of 8.6% vs 18.7%). The main drawback is a slightly lower flip rate. This is especially true for the Yelp dataset in

[8]Note that **BERT** generates counterfactuals whose average fluency is in some cases, e.g. in the OLID dataset, lower than 1. This is likely to be an artifact caused by using in this case the same model to infill the masked texts and to evaluate fluency. In other words, this means that the fluency model considers the infilled words to be more likely than the words of the original text.

Table 2: Results of RELITC on the Yelp sentence and CallMe dataset. BLEU scores are computed only against human-edited texts that are successful counterfactuals with respect to the black box. BLEU=1 for Human-edited texts by definition. Bold texts refer the best score for each metric (except Human generated). Other details as in Table 1.

	Yelp sentence						CallMe					
	Flip rate ↑	NED ↓	Fluency (≈ 1)	Content preserv. ↑	Mask Frac. ↓	BLEU ↑	Flip rate ↑	NED ↓	Fluency (≈ 1)	Content preserv. ↑	Mask Frac. ↓	BLEU ↑
HUMAN	0.766	0.588	1.750	0.616	-	1.000	0.752	0.157	1.092	0.766	-	1.000
BERT_{L2R}	0.905	0.203	**1.174**	0.700	0.159	0.534	0.453	0.313	**1.001**	0.729	0.227	0.814
$\text{BERT}_{\text{Conf}}$	0.894	0.208	1.182	0.666	0.165	0.528	0.430	0.306	1.112	0.729	0.220	0.813
MiCe	**1.000**	0.190	1.383	0.663	0.143	0.526	**0.983**	0.540	1.106	0.470	0.311	**0.816**
$\text{RELITC}_{\text{L2R}}$	0.972	**0.149****	1.545	**0.727****	0.097**	**0.542**	0.815	0.295**	1.224	**0.741****	**0.219****	0.793
$\text{RELITC}_{\text{Conf}}$	0.970	0.150**	1.556	0.726**	**0.095****	**0.542**	0.784	**0.290****	1.208	0.737**	0.227**	0.802

which our approach is successful in 94% of the cases as compared to MiCe, which is successful in more than 99% of the instances.

In Table 2 we show further comparison between **RELITC** and **MiCe** for two other datasets: Yelp sentence and CallMe. We observe the same tendencies as described above with the difference of higher averages for fluency in both cases and a significantly lower flip rate for CallMe. This latter observation might be due to the fact that it is more difficult to generate counterfactuals for this sexism detection task as can be seen from the higher mask fraction of all the models in comparison to the other tasks involving texts with comparable length (i.e., OLID and Yelp sentence whose mask fraction of RELITC is lower than 10%). In addition, the flip rate is competitive when the counterfactual label is non-sexist, while it is low when the label is sexist (97.2% vs. 75%, data not shown). In other words, RELITC struggles in transforming a non-sexist text into a sexist one. This might indicate that the fine-tuning of RELITC is less effective than the one of MiCe when the dataset is imbalanced. Indeed, only 15% of the tweets are labeled as sexist.

In these experiments we employed Integrated Gradients as the feature attribution method of RELITC but alternative methods can be used. To explore the performance of RELITC with a different feature attribution method, we conducted a small-scale analysis on a random sample of 100 examples from each dataset. By replacing Integrated Gradients with SHAP, a model-agnostic feature attribution method [22], we obtained results comparable to the ones shown in Table 1 and Table 2. It is worth to notice that the mask fraction and NED for the Yelp dataset improve (from 0.172 to 0.093 and from 0.119 to 0.084, respectively). This points to the feasibility of using diverse feature attribution methods during the fine-tuning and mask infilling stages of RELITC, including model-agnostic methods.

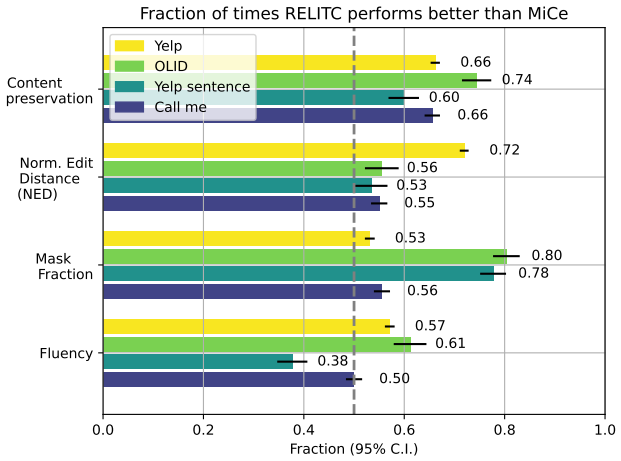

Figure 3: Fraction of times RELITC outperforms MiCe for each metric. Colors correspond to datasets, the vertical dashed line to 0.5, text labels to the observed value and horizontal black lines to 95% confidence intervals.

Pairwise comparisons: In order to further assess to what extent $\textbf{RELITC}_{\text{Conf}}$ performs differently from **MiCe**, we computed the fraction of input text instances for which $\text{RELITC}_{\text{Conf}}$ counterfactuals obtain better values for the different metrics than the ones generated by MiCe.[9] This allows us to compare the results at the level of single examples, while the scores in Table 2 are obtained by averaging the results across the dataset.

Results are shown in Figure 3 for all four datasets and metrics which are calculated at the instance level (NED, Fluency, Mask Fraction and Content preservation). Consistently with the previous considerations, $\text{RELITC}_{\text{Conf}}$ generates more frequently counterfactuals that better preserve the content (more than 60% of the times across all datasets), and that require a lower mask fraction (more than 53% of the times across all datasets). This is also true for the normalized edit distance (NED, also at least 53%) and for fluency with the exception of the Yelp sentence dataset (light blue bars).

$\text{RELITC}_{\text{Conf}}$ performs best for OLID (green bars) in terms of content preservation, mask fraction and fluency where it outperforms MiCe 74%, 80%, and 61% of the times respectively. The latter is interesting in comparison with Table 1, where the average of the fluency score was closer to one (and thus better on average) for MiCe. This suggests a high skewness of the underlying distribution. In fact, we observe a similar effect for the CallMe dataset, where the average of fluency is considerably farther from the ideal score than MiCe, but at an instance wise comparison both approaches perform equally well and outperform the other ~50% of the times.

To summarise, the **Confident** ordering contributes in generating more fluent counterfactuals compared to the **L2R** ordering, while our **RELITC** is better able at preserving the content of the original input text with a lower number of edits than **MiCe**. When compared to single data examples, our **RELITC** performs better than **MiCe**

[9] If one of the two models is not able to generate a successful counterfactual, we consider the other model to perform better. Thus, this evaluation favors the model resulting in higher flip rate. Ties are excluded and count for neither side.

in more than 50% of the examples for all the metrics, except for Fluency in Yelp Sentence and CallMe (only parity).

Manual evaluation: We conducted a human evaluation similar to the pairwise comparison discussed above to validate these results. Four annotators rated a sample of 100 tweets from the OLID dataset, with each tweet evaluated by three annotators. Annotators compared blindly and in random order the counterfactuals generated by $\text{RELITC}_{\text{Conf}}$ and MiCe for each tweet based on two criteria: fluency and content preservation. Ratings were given on a 5-point Likert scale from "Text 1 is clearly better" to "Text 2 is clearly better", with an intermediate score indicating that "Text 1 is as good as Text 2". We assessed the inter-annotator agreement via Krippendorff's α coefficient, obtaining 0.32. Although low, the agreement is far from random ($\alpha = 0$) and reflects the difficulty of the task. Indeed, the OLID dataset is composed of tweets, most of which are not grammatically correct thus making it difficult to properly assess how well the generated counterfactuals read in English. Then, we aggregated annotators' ratings by resolving conflicts with majority vote and discarded a total of 5 tweets because the predicted label of the original text was wrong. Whenever the three annotators gave three different ratings, we kept the intermediate one. The evaluation reveals that $\text{RELITC}_{\text{Conf}}$ counterfactuals better preserve the content of the input text in 81% of the cases and are more fluent 66% of the times. These findings align with the results shown in Figure 3 for the OLID dataset, where $\text{RELITC}_{\text{Conf}}$ outperforms MiCe in content preservation in 74% of cases and fluency in 61% of cases.

4.4 Comparison with human edits

Additionally, we also analyze how the generated counterfactuals for the Yelp sentence and CallMe datasets compare with counterfactuals generated by human annotators, for which these two datasets include several hundreds of instances.

We use the BLEU score to measure the similarly between the manual and machine generated counterfactuals. For the sake of fairness, we consider human-edited texts only if they are successful counterfactuals according to the output of the black box classifier trained on the corresponding dataset. This percentage can be observed in the flip-rate columns of the HUMAN row in Table 2, where the BLEU columns give the results of this analysis for the Yelp sentence and CallMe dataset. In both cases, we observe no considerable difference between the **L2R** and **Confident** orderings, regardless of the underlying model. In particular, the $\textbf{BERT}_{\text{Conf}}$ baseline performs marginally better than our RELITC in the CallMe dataset (0.813 vs 0.802), but the flip rate is almost halved (45% vs. 78%). On the other hand, **RELITC** reaches higher BLEU scores than **MiCe** in the Yelp sentence dataset (0.542 vs. 0.526), while it is the opposite for the CallMe dataset (0.802 of RELITC vs. 0.816 of MiCe).

It is also worth comparing the similarity of the human generated counterfactuals in terms of normalized edit distance (NED) and Content preservation. In the CallMe dataset these scores are the best for the human generated counterfactuals (NED = 0.157 and Content preservation = 0.766), while for Yelp sentence the opposite holds (NED = 0.588 and Content preservation = 0.616). Thus, the machine generated counterfactuals and in particular the ones of RELITC are better in full-filling the conditions introduced in §1.

Table 3: Average runtime in seconds per input of $\textbf{RELITC}_{\text{Conf}}$ and MiCe. Subscripts represent standard deviations.

	Yelp	OLID	Yelp sentence	CallMe
$\text{RELITC}_{\text{Conf}}$	$8.76_{\pm 10.28}$	$0.71_{\pm 0.45}$	$0.28_{\pm 0.07}$	$0.46_{\pm 0.29}$
MiCe	$10.05_{\pm 6.38}$	$3.47_{\pm 3.41}$	$1.09_{\pm 0.45}$	$12.05_{\pm 16.07}$

4.5 Runtime analysis

Finally, we compare the runtime of $\textbf{RELITC}_{\text{Conf}}$ and **MiCe**. We sampled a balanced sample of 100 texts from all the four datasets to generate counterfactuals using the previously trained models. Table 3 shows that $\text{RELITC}_{\text{Conf}}$ is faster than MiCe on average across all the datasets. This is likely due to the number of parameters of the conditional masked language models employed by the two methods (110M for BERT and 220M for T5), which makes it faster for $\text{RELITC}_{\text{Conf}}$ to run a forward step through the model. Regarding Yelp, $\text{RELITC}_{\text{Conf}}$ is still faster on average but the corresponding standard deviation is large compared to the difference with MiCe, thus signaling a broader distribution of the runtime.

5 CONCLUSIONS AND FUTURE WORK

We have presented RELITC, a novel approach for generating Natural Language Counterfactuals for text classifiers. By introducing a novel infilling strategy based on model confidence, RELITC fills in the most obvious tokens first and then iterates the process while incorporating information from the already infilled words. Our experimental evaluation on four datasets demonstrates that RELITC outperforms baselines and a state-of-the-art method by generating counterfactuals that are closer to the original text, better preserve the original content, and are feasible. Additionally, generated counterfactuals are also closer to human-edited counterfactuals. We furthermore show that RELITC gains as well in speed and is able to generate counterfacutals faster. The lower number of parameters of the CMLM of RELITC contributes to the runtime improvement. These properties make RELITC a compelling method for model explainability that can foster trust towards systems that become increasingly integrated into various aspects of daily life by enhancing their transparency and accountability. Potential applications include a model debugging tool for practitioners, or a model interpretability tool that can help end users to understand the decisions of black-box classifiers. Its applicability can go beyond explainability (e.g., detoxification and data augmentation).

However, there are opportunities for further improvements. The mask infilling step of RELITC can be made more flexible to delete tokens or replace masked spans with an arbitrary number of tokens. Our experiments do not include sensitivity analysis of hyperparameters, which were chosen similarly to Ref. [39], or an evaluation of how the employment of a different CMLM would affect the performance of RELITC. Regardless of this, our results are already compelling for all the datasets. Hyperparameter tuning or the employment of a different CMLM might lead to a further improvement of the metrics. Then, despite being agnostic to the choice of the feature attribution method, we have performed experiments using only Integrated Gradients, which requires to have full access to the black-box. This allowed us to fairly compare our results with MiCe, in particular for the runtime analysis. However, since an increasing number of machine learning algorithms uses proprietary

black-box classifiers build upon LLMs, fast and feasible strategies to provide understandable explanations are required to allow humans to trust in their results and the underlying algorithms. We provided preliminary results in Section 4.3 showing that a model-agnostic feature attribution method (SHAP [22]) would be competitive as well. Although we validated the results provided by the automatic metrics through a small scale manual evaluation, further research is needed to evaluate the perceived utility and user acceptance of RELITC's counterfactuals in real-world applications.

While our work contributes to auditing black-box classifiers and addressing transparency concerns, we recognize the ethical implications of generating counterfactuals, particularly in sensitive domains like toxicity detection. Indeed, RELITC provides a counterfactual explanation for a text classified as "non-toxic" by editing the input text into a text that the black box predicts as "toxic". We are aware that this mechanism can be used for malicious purposes and, in the case of toxicity detection, to generate toxic content at scale starting from normal texts, or to find examples that might circumvent automatic content moderation tools while still being toxic. We explicitly disapprove the misuse of RELITC for malicious purposes and emphasize its potential benefits in addressing problems of transparency and interpretability of tools based on LLMs.

APPENDIX A: ADDITIONAL DETAILS AND EXAMPLES OF COUNTERFACTUALS

We fine-tuned the *bert-base-uncased* pre-trained model as the CMLM for a maximum number of 10 epochs, with batch size of 128, learning rate of 5×10^{-5} with linear decay, weight decay of 0.01, and early stopping with patience of 4. We choose the final model to be the one with minimum validation loss, computed 4 times per epoch. Finally, we compute the metrics for the counterfactuals obtained with the lower mask fraction and having the lowest minimality. If this leads to multiple counterfactuals, we average their metrics.

Table A1: Example of counterfactual from the Yelp dataset. Texts in bold correspond to the edits of each the models.

Method	Yelp Example Text
Input text	tried this when i was stopping by postnet a few stores away. the place was empty at lunchtime but figured we would give it a try anyway. the staff were very friendly but the food was really uninspiring.
$\text{RELITC}_{\text{L2R}}$	tried this when i was stopping by postnet a few stores away. the place was empty at lunchtime but figured we would give it a try anyway. the staff were very friendly but the food was really **good very yummy**.
$\text{RELITC}_{\text{Conf}}$	tried this when i was stopping by postnet a few stores away. the place was empty at lunchtime but figured we would give it a try anyway. the staff were very friendly but the food was really **good with great prices**.
MiCe	tried this when i was stopping by postnet a few stores away. the place was really **quiet** but figured we would give it a try anyway. the staff were very friendly but the **ph** was really **un remarking**.

Metrics	CF. label	NED	Fluen.	Cont. Pres.	Mask Frac.
$\text{RELITC}_{\text{L2R}}$	Positive	0.073	1.192	0.956	0.031
$\text{RELITC}_{\text{Conf}}$	Positive	0.097	1.173	0.947	0.031
MiCe	Positive	0.146	1.585	0.740	0.137

Tables A1 and A2 provide example counterfactuals generated by RELITC and MiCe and the corresponding metrics. The changes with respect to the input text are highlighted in bold. Note than in some cases the tokenizer splits a single words into multiple tokens.

Table A2: Example counterfactuals of three more datasets.

Method	OLID Text
Input text	#arianaasesina? is that serious?! holy s∗it, please your fu∗ing ass∗oles, don't blame someone for the death of other one. she is sad enough for today, don't you see? it isn't fault of none, he had an overdose and died. end. stop wanting someone to blame, fu∗rs.
$\text{RELITC}_{\text{L2R}}$	#arianaasesina? is that serious?! holy **mother**, please your **young sweethearts**, don't blame someone for the death of other one. she is sad enough for today, don't you see? it isn't fault of none, he had an overdose and died. end. stop wanting someone to blame, **watchers**.
$\text{RELITC}_{\text{Conf}}$	#arianaasesina? is that serious?! holy **mother**, please your **poor sweeties**, don't blame someone for the death of other one. she is sad enough for today, don't you see? it isn't fault of none, he had an overdose and died. end. stop wanting someone to blame, **heartbreakers**.
MiCe	#arianaasesina? is that serious?! **#stop, keep** your **head high**, don't blame someone for the death of other one. she is sad enough for today, don't you see? it isn't fault of none, he had an **accident** and died. end. stop wanting someone to blame, **rabid conservatives**.

Metrics	CF. label	NED	Fluen.	Cont. Pres.	Mask Frac.
$\text{RELITC}_{\text{L2R}}$	Non-off	0.063	1.034	0.926	0.063
$\text{RELITC}_{\text{Conf}}$	Non-off	0.063	1.033	0.932	0.063
MiCe	Non-off	0.127	0.958	0.867	0.172

Method	Yelp sentence Example Text
Input text	it is very very family oriented and we loved that too!
HUMAN	too many single people, not a place for your family
$\text{RELITC}_{\text{L2R}}$	it is very very **child** oriented and we **hated** that too!
$\text{RELITC}_{\text{Conf}}$	it is very very **child** oriented and we **hated** that too!
MiCe	it is very very **cheap food** and we **hate** that too!

Metrics	CF. label	NED	Fluen.	Cont. Pres.	Mask Frac.
HUMAN	Negative	0.286	0.981	0.703	-
$\text{RELITC}_{\text{L2R}}$	Negative	0.167	1.130	0.624	0.094
$\text{RELITC}_{\text{Conf}}$	Negative	0.167	1.130	0.624	0.094
MiCe	Negative	0.250	1.060	0.090	0.206

Method	CallMe Example Text
Input text	the husband is responsible for the family so the wife must obey him.
HUMAN	the parents are responsible for the family so the children must obey them.
$\text{RELITC}_{\text{L2R}}$	the **president** is responsible for the family **and the people who** obey him.
$\text{RELITC}_{\text{Conf}}$	the **lord** is responsible for the family **and the family must** obey him.
MiCe	the **grandpa** is responsible for the family so the **fam** must **protect** him.

Metrics	CF. label	NED	Fluen.	Cont. Pres.	Mask Frac.
HUMAN	Non-sexist	0.286	0.981	0.703	-
$\text{RELITC}_{\text{L2R}}$	Non-sexist	0.286	1.245	0.512	0.313
$\text{RELITC}_{\text{Conf}}$	Non-sexist	0.214	0.641	0.743	0.250
MiCe	Non-sexist	0.214	2.389	0.303	0.172

REFERENCES

[1] Carl Allen and Timothy Hospedales. 2019. Analogies Explained: Towards Understanding Word Embeddings. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Long Beach, CA, USA, 223–231. https://proceedings.mlr.press/v97/allen19a.html

[2] David Alvarez-Melis and Tommi Jaakkola. 2017. A causal framework for explaining the predictions of black-box sequence-to-sequence models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 412–421. https://doi.org/10.18653/v1/D17-1042

[3] Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, and Hsiao-Wuen Hon. 2020. UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, Vienna, Austria, 642–652. https://proceedings.mlr.press/v119/bao20a.html

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., virtual conference, 1877–1901. https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[5] Zeming Chen, Qiyue Gao, Antoine Bosselut, Ashish Sabharwal, and Kyle Richardson. 2023. DISCO: Distilling Counterfactuals with Large Language Models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Toronto, Canada, 5514–5528. https://doi.org/10.18653/v1/2023.acl-long.302

[6] Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A Survey of the State of Explainable AI for Natural Language Processing. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, Suzhou, China, 447–459. https://aclanthology.org/2020.aacl-main.46

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/ARXIV.1810.04805

[8] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 320–335. https://doi.org/10.18653/v1/2022.acl-long.26

[9] Lilian Edwards and Michael Veale. 2017. Slave to the algorithm: Why a right to an explanation is probably not the remedy you are looking for. *Duke L. & Tech. Rev.* 16 (2017), 18.

[10] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 889–898. https://doi.org/10.18653/v1/P18-1082

[11] Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2014. Retrofitting Word Vectors to Semantic Lexicons. https://doi.org/10.48550/ARXIV.1411.4166

[12] Zee Fryer, Vera Axelrod, Ben Packer, Alex Beutel, Jilin Chen, and Kellie Webster. 2022. Flexible text generation for counterfactual fairness probing. In *Proceedings of the Sixth Workshop on Online Abuse and Harms (WOAH)*. Association for Computational Linguistics, Seattle, Washington (Hybrid), 209–229. https://aclanthology.org/2022.woah-1.20

[13] B. Fuglede and F. Topsoe. 2004. Jensen-Shannon divergence and Hilbert space embedding. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.* IEEE, Chicago, IL, USA, 31–. https://doi.org/10.1109/ISIT.2004.1365067

[14] Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H. Chi, and Alex Beutel. 2019. Counterfactual Fairness in Text Classification through Robustness. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society* (Honolulu, HI, USA) *(AIES '19)*. Association for Computing Machinery, New York, NY, USA, 219–226. https://doi.org/10.1145/3306618.3317950

[15] Reza Ghaeini, Xiaoli Fern, and Prasad Tadepalli. 2018. Interpreting Recurrent and Attention-Based Neural Models: a Case Study on Natural Language Inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4952–4957. https://doi.org/10.18653/v1/D18-1537

[16] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The Curious Case of Neural Text Degeneration. https://doi.org/10.48550/ARXIV.1904.09751

[17] Di Jin, Zhijing Jin, Zhiting Hu, Olga Vechtomova, and Rada Mihalcea. 2022. Deep Learning for Text Style Transfer: A Survey. *Computational Linguistics* 48, 1 (04 2022), 155–205. https://doi.org/10.1162/coli_a_00426 arXiv:https://direct.mit.edu/coli/article-pdf/48/1/155/2006608/coli_a_00426.pdf

[18] Mark T. Keane and Barry Smyth. 2020. Good Counterfactuals and Where to Find Them: A Case-Based Technique for Generating Counterfactuals for Explainable AI (XAI). In *Case-Based Reasoning Research and Development*, Ian Watson and Rosina Weber (Eds.). Springer International Publishing, Cham, 163–178.

[19] Piyawat Lertvittayakumjorn and Francesca Toni. 2021. Explanation-Based Human Debugging of NLP Models: A Survey. *Transactions of the Association for Computational Linguistics* 9 (12 2021), 1508–1528. https://doi.org/10.1162/tacl_a_00440 arXiv:https://direct.mit.edu/tacl/article-pdf/doi/10.1162/tacl_a_00440/1983435/tacl_a_00440.pdf

[20] Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, Retrieve, Generate: a Simple Approach to Sentiment and Style Transfer. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1865–1874. https://doi.org/10.18653/v1/N18-1169

[21] Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically Masked Language Model Capable of Autoregressive Generation in Arbitrary Word Order. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 263–274. https://doi.org/10.18653/v1/2020.acl-main.24

[22] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc., Long Beach, CA, USA. https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf

[23] Nishtha Madaan, Inkit Padhi, Naveen Panwar, and Diptikalyan Saha. 2021. Generate Your Counterfactuals: Towards Controlled Counterfactual Generation for Text. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 15 (May 2021), 13516–13524. https://doi.org/10.1609/aaai.v35i15.17594

[24] Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-Hoc Interpretability for Neural NLP: A Survey. *ACM Comput. Surv.* 55, 8, Article 155 (dec 2022), 42 pages. https://doi.org/10.1145/3546577

[25] Binny Mathew, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee. 2021. HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 17 (May 2021), 14867–14875. https://doi.org/10.1609/aaai.v35i17.17745

[26] Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence* 267 (2019), 1–38. https://doi.org/10.1016/j.artint.2018.07.007

[27] John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. TextAttack: A Framework for Adversarial Attacks, Data Augmentation, and Adversarial Training in NLP. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Association for Computational Linguistics, Online, 119–126. https://doi.org/10.18653/v1/2020.emnlp-demos.16

[28] James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable Prediction of Medical Codes from Clinical Text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1101–1111. https://doi.org/10.18653/v1/N18-1100

[29] Cathy O'Neil. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown Publishing Group, USA.

[30] Abhishek Panigrahi, Harsha Vardhan Simhadri, and Chiranjib Bhattacharyya. 2019. Word2Sense: Sparse Interpretable Word Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 5692–5705. https://doi.org/10.18653/v1/P19-1570

[31] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, 311–318. https://doi.org/10.3115/1073083.1073135

[32] Matthew E. Peters, Mark Neumann, Luke Zettlemoyer, and Wen-tau Yih. 2018. Dissecting Contextual Word Embeddings: Architecture and Representation. https://doi.org/10.48550/ARXIV.1808.08949

[33] Nina Poerner, Hinrich Schütze, and Benjamin Roth. 2018. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 340–350. https://doi.org/10.18653/v1/P18-1032

[34] Chen Qian, Fuli Feng, Lijie Wen, Chunping Ma, and Pengjun Xie. 2021. Counterfactual Inference for Text Classification Debiasing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 5434–5445. https://doi.org/10.18653/v1/2021.acl-long.422

[35] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67. http://jmlr.org/papers/v21/20-074.html

[36] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. https://doi.org/10.48550/ARXIV.1908.10084

[37] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, California, USA) *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 1135–1144. https://doi.org/10.1145/2939672.2939778

[38] Marcel Robeer, Floris Bex, and Ad Feelders. 2021. Generating Realistic Natural Language Counterfactuals. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, Punta Cana, Dominican Republic, 3611–3625. https://doi.org/10.18653/v1/2021.findings-emnlp.306

[39] Alexis Ross, Ana Marasović, and Matthew Peters. 2021. Explaining NLP Models via Minimal Contrastive Editing (MiCE). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Association for Computational Linguistics, Online, 3840–3852. https://doi.org/10.18653/v1/2021.findings-acl.336

[40] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. 2020. Masked Language Model Scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Online, 2699–2712. https://doi.org/10.18653/v1/2020.acl-main.240

[41] Mattia Samory, Indira Sen, Julian Kohne, Fabian Flöck, and Claudia Wagner. 2021. "Call me sexist, but..." : Revisiting Sexism Detection Using Psychological Scales and Adversarial Samples. *Proceedings of the International AAAI Conference on Web and Social Media* 15, 1 (May 2021), 573–584. https://doi.org/10.1609/icwsm.v15i1.18085

[42] Tianxiao Shen, Victor Quach, Regina Barzilay, and Tommi Jaakkola. 2020. Blank Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 5186–5198. https://doi.org/10.18653/v1/2020.emnlp-main.420

[43] Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. Transforming Delete, Retrieve, Generate Approach for Controlled Text Style Transfer. https://doi.org/10.48550/ARXIV.1908.09368

[44] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic Attribution for Deep Networks. https://doi.org/10.48550/ARXIV.1703.01365

[45] Alona Sydorova, Nina Poerner, and Benjamin Roth. 2019. Interpretable Question Answering on Knowledge Bases and Text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 4943–4951. https://doi.org/10.18653/v1/P19-1488

[46] Victor Veitch, Alexander D'Amour, Steve Yadlowsky, and Jacob Eisenstein. 2021. Counterfactual Invariance to Spurious Correlations: Why and How to Pass Stress Tests. https://doi.org/10.48550/ARXIV.2106.00545

[47] Sandra Wachter, Brent Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harv. JL & Tech.* 31 (2017), 841.

[48] Eric Wallace, Shi Feng, and Jordan Boyd-Graber. 2018. Interpreting Neural Networks with Nearest Neighbors. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics, Brussels, Belgium, 136–144. https://doi.org/10.18653/v1/W18-5416

[49] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel Weld. 2021. Polyjuice: Generating Counterfactuals for Explaining, Evaluating, and Improving Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Online, 6707–6723. https://doi.org/10.18653/v1/2021.acl-long.523

[50] Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. "Mask and Infill" : Applying Masked Language Model to Sentiment Transfer. https://doi.org/10.48550/ARXIV.1908.08039

[51] Yuexiang Xie, Fei Sun, Yang Deng, Yaliang Li, and Bolin Ding. 2021. Factual Consistency Evaluation for Text Summarization via Counterfactual Estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. Association for Computational Linguistics, Punta Cana, Dominican Republic, 100–110. https://doi.org/10.18653/v1/2021.findings-emnlp.10

[52] Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Predicting the Type and Target of Offensive Posts in Social Media. https://doi.org/10.48550/ARXIV.1902.09666

[53] Wei Emma Zhang, Quan Z. Sheng, Ahoud Alhazmi, and Chenliang Li. 2020. Adversarial Attacks on Deep-Learning Models in Natural Language Processing: A Survey. *ACM Trans. Intell. Syst. Technol.* 11, 3, Article 24 (apr 2020), 41 pages. https://doi.org/10.1145/3374217

[54] Julia El Zini and Mariette Awad. 2022. On the Explainability of Natural Language Processing Deep Models. *ACM Comput. Surv.* 55, 5, Article 103 (dec 2022), 31 pages. https://doi.org/10.1145/3529755