Explaining NLP Models via Minimal Contrastive Editing (MICE)

Alexis Ross[†] **Ana Marasović**[†◇] **Matthew E. Peters**[†]

[†]Allen Institute for Artificial Intelligence, Seattle, WA, USA
[◇]Paul G. Allen School of Computer Science and Engineering, University of Washington
{alexisr,anam,matthewp}@allenai.org

Abstract

Humans have been shown to give contrastive explanations, which explain why an observed event happened *rather than* some other counterfactual event (the *contrast case*). Despite the influential role that contrastivity plays in how humans explain, this property is largely missing from current methods for explaining NLP models. We present MINIMAL CONTRASTIVE EDITING (MICE), a method for producing contrastive explanations of model predictions in the form of edits to inputs that change model outputs to the contrast case. Our experiments across three tasks—binary sentiment classification, topic classification, and multiple-choice question answering—show that MICE is able to produce edits that are not only contrastive, but also *minimal* and *fluent*, consistent with human contrastive edits. We demonstrate how MICE edits can be used for two use cases in NLP system development—debugging incorrect model outputs and uncovering dataset artifacts—and thereby illustrate that producing contrastive explanations is a promising research direction for model interpretability.

1 Introduction

Cognitive science and philosophy research has shown that human explanations are **contrastive** (Miller, 2019): People explain why an observed event happened rather than some counterfactual event called the *contrast case*. This contrast case plays a key role in modulating what explanations are given. Consider Figure 1. When we seek an explanation of the model's prediction "by train," we seek it not in absolute terms, but in contrast to another possible prediction (i.e. "on foot"). Additionally, we tailor our explanation to this contrast case. For instance, we might explain why the prediction is "by train" and not "on foot" by saying that the writer discusses meeting Ann at the train station instead of at Ann's home on foot; such information is captured by the edit (bolded red) that results in the new model prediction "on foot." For a different contrast prediction, such as "by car," we would provide a different explanation. In this work, we propose to give contrastive explanations of model predictions in the form of targeted minimal edits, as shown in Figure 1, that cause the model to change its original prediction to the contrast prediction.

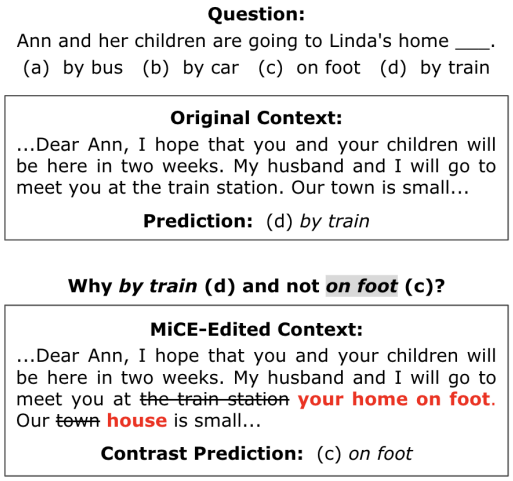

Figure 1: An example MICE edit for a multiple-choice question from the RACE dataset. MICE generates contrastive explanations in the form of edits to inputs that change model predictions to target (contrast) predictions. The edit (bolded in red) is minimal and fluent, and it changes the model's prediction from "by train" to the contrast prediction "on foot" (highlighted in gray).

Given the key role that contrastivity plays in human explanations, making model explanations contrastive could make them more user-centered and thus more useful for their intended purposes, such as debugging and exposing dataset biases (Ribera and Lapedriza, 2019)—purposes which require that *humans* work with explanations (Alvarez-Melis et al., 2019). However, many currently popular instance-based explanation methods produce

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3840–3852

highlights—segments of input that support a prediction (Zaidan et al., 2007; Lei et al., 2016; Chang et al., 2019; Bastings et al., 2019; Yu et al., 2019; DeYoung et al., 2020; Jain et al., 2020; Belinkov and Glass, 2019) that can be derived through gradients (Simonyan et al., 2014; Smilkov et al., 2017; Sundararajan et al., 2017), approximations with simpler models (Ribeiro et al., 2016), or attention (Wiegreffe and Pinter, 2019; Sun and Marasović, 2021). These methods are not contrastive, as they leave the contrast case undetermined; they do not tell us what would have to be different for a model to have predicted a particular contrast label.[1]

As an alternative approach to NLP model explanation, we introduce **MINIMAL CONTRASTIVE EDITING (MICE)**—a two-stage approach to generating contrastive explanations in the form of targeted minimal edits (as shown in Figure 1). Given an input, a fixed PREDICTOR model, and a contrast prediction, MICE generates edits to the input that change the PREDICTOR's output from the original prediction to the contrast prediction. We formally define our edits and describe our approach in §2.

We design MICE to produce edits with properties motivated by human contrastive explanations. First, we desire edits to be **minimal**, altering only small portions of input, a property which has been argued to make explanations more intelligible (Alvarez-Melis et al., 2019; Miller, 2019). Second, MICE edits should be **fluent**, resulting in text natural for the domain and ensuring that any changes in model predictions are not driven by inputs falling out of distribution of naturally occurring text. Our experiments (§3) on three English-language datasets, IMDB, NEWSGROUPS, and RACE, validate that MICE edits are indeed contrastive, minimal, and fluent.

We also analyze the quality of MICE edits (§4) and show how they may be used for two use cases in NLP system development. First, we show that MICE edits are comparable in size and fluency to human edits on the IMDB dataset. Next, we illustrate how MICE edits can facilitate debugging individual model predictions. Finally, we show how MICE edits can be used to uncover dataset artifacts learned by a powerful PREDICTOR model.[2]

[1]Free-text rationales (Narang et al., 2020) can be contrastive if human justifications are collected by asking "why... instead of..." which is not the case with current benchmarks (Camburu et al., 2018; Rajani et al., 2019; Zellers et al., 2019).

[2]Our code and trained EDITOR models are publicly available at https://github.com/allenai/mice.

2 MICE: Minimal Contrastive Editing

This section describes our proposed method, MINIMAL CONTRASTIVE EDITING, or MICE, for explaining NLP models with contrastive edits.

2.1 MICE Edits as Contrastive Explanations

Contrastive explanations are answers to questions of the form *Why p and not q?* They explain why the observed event p happened instead of another event q, called the *contrast case*.[3] A long line of research in the cognitive sciences and philosophy has found that human explanations are contrastive (Van Fraassen, 1980; Lipton, 1990; Miller, 2019). Human contrastive explanations have several hallmark characteristics. First, they cite *contrastive features*: features that result in the contrast case when they are changed in a particular way (Chin-Parker and Cantelon, 2017). Second, they are minimal in the sense that they rarely cite the entire causal chain of a particular event, but select just a few relevant causes (Hilton, 2017). In this work, we argue that a minimal edit to a model input that causes the model output to change to the contrast case has both these properties and can function as an effective contrastive explanation. We first give an illustration of contrastive explanations humans might give and then show how minimal contrastive edits offer analogous contrastive information.

As an example, suppose we want to explain why the answer to the question "*Q*: Where can you find a clean pillow case that is not in use?" is "*A*: the drawer."[4] If someone asks why the answer is not "*C1*: on the bed," we might explain: "*E1*: Because only the drawer stores pillow cases that are not in use." However, *E1* would *not* be an explanation of why the answer is not "*C2*: in the laundry hamper," since both drawers and laundry hampers store pillow cases that are not in use. For contrast case *C2*, we might instead explain: "*E2*: Because only laundry hampers store pillow cases that are not clean." We cite different parts of the original question depending on the contrast case.

In this work, we propose to offer contrastive explanations in the form of minimal edits that result in the contrast case as model output. Such edits are effective contrastive explanations because, by construction, they highlight contrastive features. For

[3]Related work also calls it the *foil* (Miller, 2019).

[4]Inspired by an example in Talmor et al. (2019): Question: "Where would you store a pillow case that is not in use?" Choices: "drawer, kitchen cupboard, bedding store, england."

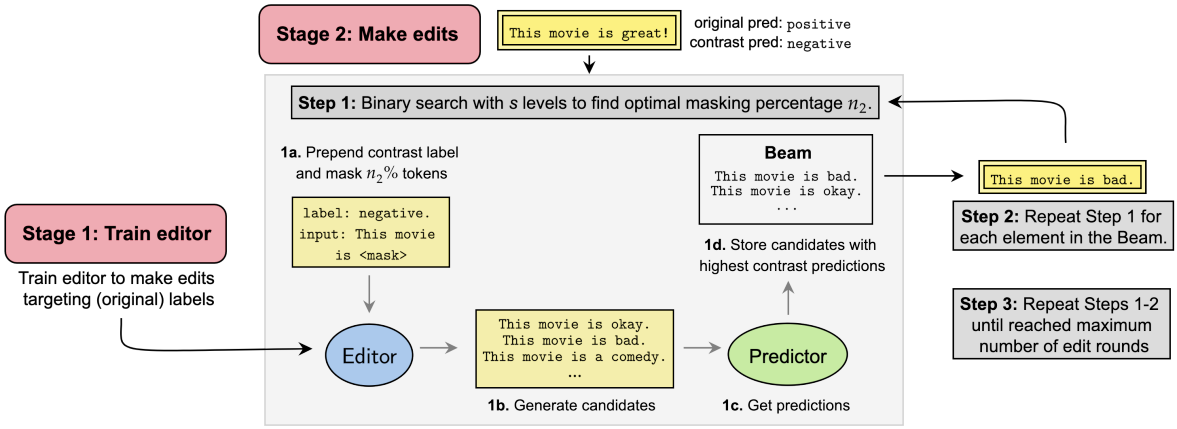

Figure 2: An overview of MICE, our two-stage approach to generating edits. In Stage 1 (§2.3), we train the EDITOR to make edits targeting specific predictions from the PREDICTOR. In Stage 2 (§2.4), we make contrastive edits with the EDITOR model from Stage 1 such that the PREDICTOR changes its output to the contrast prediction.

example, a contrastive edit of the original question for contrast case *C1* would be: "Where can you find a clean pillow case that is ~~**not**~~ in use?"; the information provided by this edit—that it is whether or not the pillow case is in use that determines whether the answer is "the drawer" or "on the bed"—is analogous to the information provided by *E1*. Similarly, a contrastive edit for contrast case *C2* that changed the question to "Where can you find a ~~**clean**~~ **dirty** pillow case that is not in use?" provides analogous information to *E2*.

2.2 Overview of MICE

We define a contrastive edit to be a modification of an input instance that causes a PREDICTOR model (whose behavior is being explained) to change its output from its original prediction for the unedited input to a given target (contrast) prediction. Formally, for textual inputs, given a fixed PREDICTOR f, input $\boldsymbol{x} = (x_1, x_2, ..., x_N)$ of N tokens, original prediction $f(\boldsymbol{x}) = y_p$ and contrast prediction $y_c \neq y_p$, a contrastive edit is a mapping $e : (x_1, ..., x_N) \rightarrow (x'_1, ..., x'_M)$ such that $f(e(\boldsymbol{x})) = y_c$.

We propose MICE, a two-stage approach to generating contrastive edits, illustrated in Figure 2. In Stage 1, we prepare a highly-contextualized EDITOR model to associate edits with given end-task labels (i.e., labels for the task of the PREDICTOR) such that the contrast label y_c is not ignored in MICE's second stage. Intuitively, we do this by masking the spans of text that are "important" for the given target label (as measured by the PREDICTOR's gradients) and training our EDITOR to reconstruct these spans of text given the masked text and target label as input. In Stage 2 of MICE, we generate contrastive edits $e(\boldsymbol{x})$ using the EDITOR model from Stage 1. Specifically, we generate candidate edits $e(\boldsymbol{x})$ by masking different percentages of $\boldsymbol{x}$ and giving masked inputs with prepended contrast label y_c to the EDITOR; we use binary search to find optimal masking percentages and beam search to keep track of candidate edits that result in the highest probability of the contrast labels $p(y_c|e(\boldsymbol{x}))$ given by the PREDICTOR.

2.3 Stage 1: Fine-tuning the EDITOR

In Stage 1 of MICE, we fine-tune the EDITOR to infill masked *spans* of text in a targeted manner. Specifically, we fine-tune a pretrained model to infill masked spans given masked text and a target end-task label as input. In this work, we use the TEXT-TO-TEXT TRANSFER TRANSFORMER (T5) model (Raffel et al., 2020) as our pretrained EDITOR, but any model suitable for span infilling can in principle be the EDITOR in MICE. The addition of the target label allows the highly-contextualized EDITOR to condition its predictions on both the masked context and the given target label such that the contrast label is not ignored in Stage 2. What to use as target labels during Stage 1 depends on who the end-users of MICE are. The end-user could be: (1) a model developer who has access to the labeled data used to train the predictor, or (2) lay-users, domain experts, or other developers without access to the labeled data. In the former case, we could use the gold label as targets, and in the latter case, we could use the labels predicted by PREDICTOR. Therefore, during fine-tuning, we experiment with using both gold labels and original predictions

y_p of our PREDICTOR model as target labels. To provide target labels, we prepend them to inputs to the EDITOR. For more information about how these inputs are formatted, see Appendix B. Results in Table 2 show that fine-tuning with target labels results in better edits than fine-tuning without them.

The above procedure allows our EDITOR to condition its infilled spans on both the context and the target label. But this still leaves open the question of where to mask our text. Intuitively, we want to mask the tokens that contribute most to the PREDICTOR's predictions, since these are the tokens that are most strongly associated with the target label. We propose to use gradient attribution (Simonyan et al., 2014) to choose tokens to mask. For each instance, we take the gradient of the predicted logit for the target label with respect to the embedding layers of f and take the ℓ_1 norm across the embedding dimension. We then mask the $n_1\%$ of tokens with the highest gradient norms. We replace consecutive tokens (i.e., spans) with sentinel tokens, following Raffel et al. (2020). Results in Table 1 show that gradient-based masking outperforms random masking.

2.4 Stage 2: Making Edits with the EDITOR

In the second stage of our approach, we use our fine-tuned EDITOR to make edits using beam search (Reddy, 1977). In each round of edits, we mask consecutive spans of $n_2\%$ of tokens in the original input, prepend the contrast prediction to the masked input, and feed the resulting masked instance to the EDITOR; the EDITOR then generates m edits. The masking procedure during this stage is gradient-based as in Stage 1.

In one round of edits, we conduct a binary search with s levels over values of n_2 between values $n_2 = 0\%$ to $n_2 = 55\%$ to efficiently find a value of n_2 that is large enough to result in the contrast prediction while also modifying only minimal parts of the input. After each round of edits, we get f's predictions on the edited inputs, order them by contrast prediction probabilities, and update the beam to store the top b edited instances. As soon as an edit $e^* = e(\boldsymbol{t})$ is found that results in the contrast prediction, i.e., $f(e^*) = y_c$, we stop the search procedure and return this edit. For generation, we use a combination of top-k (Fan et al., 2018) and top-p (nucleus) sampling (Holtzman et al., 2020).[5]

[5]We use this combination because we observed in preliminary experiments that it led to good results.

3 Evaluation

This section presents empirical findings that MICE produces minimal and fluent contrastive edits.

3.1 Experimental Setup

Tasks We evaluate MICE on three English-language datasets: IMDB, a binary sentiment classification task (Maas et al., 2011), a 6-class version of the 20 NEWSGROUPS topic classification task (Lang, 1995), and RACE, a multiple choice question-answering task (Lai et al., 2017).[6]

PREDICTORS MICE can be used to make contrastive edits for any differentiable PREDICTOR model, i.e., any end-to-end neural model. In this paper, for each task, we train a PREDICTOR model f built on ROBERTA-LARGE (Liu et al., 2019), and fix it during evaluation. The test accuracies of our PREDICTORS are 95.9%, 85.3% and 84% for IMDB, NEWSGROUPS, and RACE, respectively. For training details, see Appendix A.1.

EDITORS Our EDITORS build on the base version of T5. For fine-tuning our EDITORS (Stage 1), we use the original training data used to train PREDICTORS. We randomly split the data, 75%/25% for fine-tuning/validation and fine-tune until the validation loss stops decreasing (for a max of 10 epochs) with $n_1\%$ of tokens masked, where n_1 is a randomly chosen value in $[20, 55]$. For more details, see Appendix A.2. In Stage 2, for each instance, we set the label with the second highest predicted probability as the contrast prediction. We set beam width $b = 3$, consider $s = 4$ search levels during binary search over n_2 in each edit round, and run our search for a max of 3 edit rounds. For each n_2, we sample $m = 15$ generations from our fine-tuned EDITORS with $p = 0.95$, $k = 30$.[7]

Metrics We evaluate MICE on the test sets of the three datasets. The RACE and NEWSGROUPS test sets contain 4,934 and 7,307 instances, respectively.[8] For IMDB, we randomly sample 5K of the

[6]We create this 6-class version by mapping the 20 existing subcategories to their respective larger categories—i.e. "talk.politics.guns" and "talk.religion.misc" → "talk." We do this in order to make the label space smaller. The resulting classes are: alt, comp, misc, rec, sci, and talk.

[7]We tune these hyperparameters on a 50-instance subset of the IMDB validation set prior to evaluation. We note that for larger values of n_2, the generations produced by the T5 EDITORS sometimes degenerate; see Appendix C for details.

[8]For the NEWSGROUPS test set, there are 7,307 instances remaining after filtering out empty strings.

MICE VARIANT	IMDB ↑ Flip Rate	IMDB ↓ Minim.	IMDB ≈ 1 Fluen.	NEWSGROUPS ↑ Flip Rate	NEWSGROUPS ↓ Minim.	NEWSGROUPS ≈ 1 Fluen.	RACE ↑ Flip Rate	RACE ↓ Minim.	RACE ≈ 1 Fluen.
***PRED + GRAD**	**1.000**	**0.173**	**0.981**	**0.992**	**0.261**	0.968	0.915	**0.331**	**0.981**
***GOLD + GRAD**	**1.000**	0.185	0.979	**0.992**	0.271	0.966	**0.945**	0.335	0.979
PRED + RAND	**1.000**	0.257	0.958	0.968	0.378	0.928	0.799	0.440	0.953
GOLD + RAND	**1.000**	0.302	0.952	0.965	0.370	0.929	0.801	0.440	0.955
NO-FINETUNE	0.995	0.360	0.960	0.941	0.418	0.938	–	–	–

Table 1: Efficacy of the MICE procedure. We evaluate MICE edits on three metrics (described in §3.1): flip rate, minimality, and fluency. We report mean values for minimality and fluency. * marks full MICE variants; others explore ablations. For each property (i.e., column), the best value across MICE variants is bolded. We experiment with PREDICTOR's predictions (PRED) and gold labels (GOLD) as target labels during Stage 1. Across datasets, our GRAD MICE procedure achieves a high flip rate with small and fluent edits.

25K instances in the test set for evaluation because of the computational demands of evaluation.[9]

For each dataset, we measure the following three properties: (1) **flip rate:** the proportion of instances for which an edit results in the contrast label; (2) **minimality:** the "size" of the edit as measured by the word-level Levenshtein distance between the original and edited input, which is the minimum number of deletions, insertions, or substitutions required to transform one into the other. We report a normalized version of this metric with a range from 0 to 1—the Levenshtein distance divided by the number of words in the original input; (3) **fluency:** a measure of how similarly distributed the edited output is to the original data. We evaluate fluency by comparing masked language modeling loss on both the original and edited inputs using a pretrained model. Specifically, given the original N-length sequence, we create N copies, each with a different token replaced by a mask token, following Salazar et al. (2020). We then take a pretrained T5-BASE model and compute the average loss across these N copies. We compute this loss value for both the original input and edited input and report their *ratio*—i.e., edited / original. We aim for a value of 1.0, which indicates equivalent losses for the original and edited texts. When MICE finds multiple edits, we report metrics for the edit with the smallest value for minimality.

3.2 Results

Results are shown in Table 1. Our proposed GRAD MICE procedure (upper part of Table 1) achieves a high flip rate across all three tasks. This is the outcome regardless of whether predicted target labels (first row, 91.5–100% flip rate) or gold target labels (second row, 94.5–100% flip rate) are used for fine-tuning in Stage 1. We observe a slight improvement from using the gold labels for the RACE PREDICTOR, which may be explained by the fact that it is less accurate (with a training accuracy of 89.9%) than the IMDB and NEWSGROUPS classifiers.

MICE achieves a high flip-rate while its edits remain small and result in fluent text. In particular, MICE on average changes 17.3–33.1% of the original tokens when predicted labels are used in Stage 1 and 18.5–33.5% with gold labels. Fluency is close to 1.0 indicating no notable change in mask language modeling loss after the edit—i.e., edits fall in distribution of the original data. We achieve the best results across metrics on the IMDB dataset, as expected since IMDB is a binary classification task with a small label space. These results demonstrate that MICE presents a promising research direction for the generation of contrastive explanations; however, there is still room for improvement, especially for more challenging tasks such as RACE.

In the rest of this section, we provide results from several ablation experiments.

Fine-tuning vs. No Fine-tuning We investigate the effect of fine-tuning (Stage 1) with a baseline that skips Stage 1 altogether. For this NO-FINETUNE baseline variant of MICE, we use the vanilla pretrained T5-BASE as our EDITOR. As shown in Table 1, the NO-FINETUNE variant underperforms all other (two-stage) variants of MICE for the IMDB and NEWSGROUPS datasets.[10] Fine-

[9] A single contrastive edit is expensive and takes an average of ≈ 15 seconds per IMDB instance (≈ 230 tokens). Calculating the fluency metric adds an additional average of ≈ 16.5 seconds per IMDB instance. For more details, see Section 5.

[10] We leave RACE out from our evaluation with the NO-FINETUNE baseline because we observe that the pretrained

IMDB Condition		↑	↓	≈ 1
Stage 1	**Stage 2**	**Flip Rate**	**Minim.**	**Fluen.**
No Label	No Label	0.994	0.369	0.966
No Label	Label	0.997	0.362	0.967
Label	No Label	0.999	0.327	0.968
Label	Label	**1.000**	**0.173**	**0.981**

Table 2: Effect of using target end-task labels during the two stages of PRED+GRAD MICE on the IMDB dataset. When end-task labels are provided, they are original PREDICTOR labels during Stage 1 and contrast labels during Stage 2. The best values for each property (column) are bolded. Using end-task labels during both Stage 1 (EDITOR fine-tuning) and Stage 2 (making edits) of MICE outperforms all other conditions.

tuning particularly improves the minimality of edits, while leaving the flip rate high. We hypothesize that this effect is due to the effectiveness of Stage 2 of MICE at finding contrastive edits: Because we iteratively generate many candidate edits using beam search, we are likely to find a prediction-flipping edit. Fine-tuning allows us to find such an edit at a lower masking percentage.

Gradient vs. Random Masking We study the impact of using gradient-based masking in Stage 1 of the MICE procedure with a RAND variant, which masks spans of randomly chosen tokens. As shown in the middle part of Table 1, gradient-based masking outperforms random masking when using both predicted and gold labels across all three tasks and metrics, suggesting that the gradient-based attribution used to mask text during Stage 1 of MICE is an important part of the procedure. The differences are especially notable for RACE, which is the most challenging task according to our metrics.

Targeted vs. Un-targeted Infilling We investigate the effect of using target labels in both stages of MICE by experimenting with removing target labels during Stage 1 (EDITOR fine-tuning) and Stage 2 (making edits). As shown in Table 2, we observe that giving target labels to our EDITORS during both stages of MICE improves edit quality. Fine-tuning EDITORS without labels in Stage 1 ("No Label") leads to worse flip rate, minimality, and fluency than does fine-tuning EDITORS with labels ("Label"). Minimality is particularly affected, and we hypothesize that using target end-task labels in both stages provides signal that allows the EDITOR in Stage 2 to generate prediction-flipping edits at lower masking percentages.

4 Analysis of Edits

In this section, we compare MICE edits with human contrastive edits. Then, we turn to a key motivation for this work: the potential for contrastive explanations to assist in NLP system development. We show how MICE edits can be used to debug incorrect predictions and uncover dataset artifacts.

4.1 Comparison with Human Edits

We ask whether the contrastive edits produced by MICE are minimal and fluent in a meaningful sense. In particular, we compare these two metrics for MICE edits and human contrastive edits. We work with the IMDB contrast set created by Gardner et al. (2020), which consists of original test inputs and human-edited inputs that cause a change in *true* label. We report metrics on the subset of this contrast set for which the human-edited inputs result in a change in model prediction for our IMDB PREDICTOR; this subset consists of 76 instances. The flip rate of MICE edits on this subset is 100%. The mean minimality values of human and MICE edits are **0.149** (human) and **0.179** (MICE), and the mean fluency values are **1.01** (human) and **0.949** (MICE). The similarity of these values suggests that MICE edits are comparable to human contrastive edits along these dimensions.

We also ask to what extent human edits overlap with MICE edits. For each input, we compute the overlap between the original tokens changed by humans and the original tokens edited by MICE. The mean number of overlapping tokens, normalized by the number of original tokens edited by humans, is **0.298**. Thus, while there is some overlap between MICE and human contrastive edits, they generally change different parts of text.[11] This analysis suggests that there may exist multiple informative contrastive edits for a single input. Future work can investigate and compare the different kinds of insight that can be obtained through human and model-driven contrastive edits.

4.2 Use Case 1: Debugging Incorrect Outputs

Here, we illustrate how MICE edits can be used to debug incorrect model outputs. Consider the RACE

T5 model does not generate text formatted as span infills; we hypothesize that this model has not been trained to generate infills for masked inputs formatted as multiple choice inputs.

[11] MICE edits explain PREDICTORS' behavior and therefore need not be similar to human edits, which are designed to change gold labels.

IMDB	**Original pred** y_p = positive **Contrast pred** y_c = negative An interesting pairing of stories, this little flick manages to bring together seemingly different characters and story lines all in the backdrop of WWII and succeeds in tying them together without losing the audience. I was impressed by the depth portrayed by the different characters and also by how much I really felt I understood them and their motivations, even though the time spent on the development of each character was very limited. The outstanding acting abilities of the individuals involved with this picture are easily noted. A fun, stylized movie with a slew of comic moments and a bunch more head shaking events. ~~7/10~~ **4/10**
RACE	**Question:** Mark went up in George's plane _____. (a) twice (b) only once (c) several times (d) once or twice. **Original pred** y_p = (a) twice **Contrast pred** y_c = (b) only once When George was thirty-five, he bought a small plane and learned to fly it. He soon became very good and made his plane do all kinds of tricks. George had a friend, whose name was Mark. One day George offered to take Mark up in his plane. Mark thought, "I've traveled in a big plane several times, but I've never been in a small one, so I'll go." They went up, and George flew around for half an hour and did all kinds of tricks in the air. When they came down again, Mark was glad to be back safely, and he said to his friend in a shaking voice, "Well, George, thank you very much for those two ~~trips~~ **tricks** in your plane." George was very surprised and said, "Two ~~trips?~~ **tricks.**" ~~Yes,~~ **That's** my first ~~and my last~~ **time, George**." ~~answered~~ **said** Mark.

Table 3: Examples of edits produced by MICE. Insertions are bolded in red. Deletions are struck through. y_p is the PREDICTOR's original prediction, and y_c the contrast prediction. True labels for original inputs are underlined.

input in Table 3, for which the RACE PREDICTOR gives an incorrect prediction. In this case, a model developer may want to understand why the model got the answer wrong. This setting naturally brings rise to a contrastive question, i.e., *Why did the model predict the wrong choice ("twice") instead of the correct one ("only once")?*

The MICE edit shown offers insight into this question: Firstly, it highlights which part of the paragraph has an influence on the model prediction—the last few sentences. Secondly, it reveals that a source of confusion is Mark's joke about having traveled in George's plane twice, as changing Mark's dialogue from talking about a "first and...last" trip to a single trip results in a correct model prediction.

MICE edits can also be used to debug model *capabilities* by offering hypotheses about "bugs" present in models: For instance, the edit in Table 3 might prompt a developer to investigate whether this PREDICTOR lacks non-literal language understanding capabilities. In the next section, we show how insight from individual MICE edits can be used to uncover a bug in the form of a dataset-level artifact learned by a model. In Appendix D, we further analyze the debugging utility of MICE edits with a PREDICTOR *designed* to contain a bug.

4.3 Use Case 2: Uncovering Dataset Artifacts

Manual inspection of some edits for IMDB suggests that the IMDB PREDICTOR has learned to rely heavily on numerical ratings. For instance, in the IMDB example in Table 3, the MICE edit results in a negative prediction from the PREDICTOR even though the edited text is overwhelmingly positive. We test this hypothesis by investigating whether numerical tokens are more likely to be edited by MICE.

y_c = ***positive***		y_c = ***negative***	
Removed	**Inserted**	**Removed**	**Inserted**
4/10	excellent	10/10	awful
ridiculous	enjoy	8/10	disappointed
horrible	amazing	7/10	1
4	entertaining	9	4
predictable	10	enjoyable	annoying

Table 4: Top 5 IMDB tokens edited by MICE at a higher rate than expected given their original frequency (§4.3). Results are separated by contrast predictions.

We analyze the edits produced by MICE (GOLD + GRAD) described in §3.1. We limit our analysis to a subset of the 5K instances for which the edit produced by MICE has a minimality value of $\leq$0.05, as we are interested in finding simple artifacts driving the predictions of the IMDB PREDICTOR; this subset has 902 instances. We compute three metrics for each unique token, i.e., type t:

$$p(t) = \#\_\text{occurrences}(t) / \#\_\text{all\_tokens},$$
$$p_r(t) = \#\_\text{removals}(t) / \#\_\text{all\_removals},$$
$$p_i(t) = \#\_\text{insertions}(t) / \#\_\text{all\_insertions},$$

and report the tokens with the highest values for the ratios $p_r(t)/p(t)$ and $p_i(t)/p(t)$. Intuitively, these tokens are removed/inserted at a higher rate than expected given the frequency with which they appear in the original IMDB inputs. We exclude

tokens that occur <10 times from our analysis.

Results from this analysis are shown in Table 4. In line with our hypothesis, we observe a bias towards removing low numerical ratings and inserting high ratings when the contrast prediction y_c is positive, and vice versa when y_c is negative. In other words, in the presence of a numerical score, the PREDICTOR may ignore the content of the review and base its prediction solely on the score (as in the IMDB example in Table 3).

5 Discussion

In this section, we reflect on MICE's shortcomings. Foremost, MICE is computationally expensive. Stage 1 requires fine-tuning a large pretrained generation model as the EDITOR. More significantly, Stage 2 requires multiple rounds of forward and backward passes to find a minimal edit: Each edit round in Stage 2 requires $b \times s \times m$ decoded sequences with the EDITOR, as well as $b \times s \times m$ forward passes and b backward passes with the PREDICTOR (with $b = 1$ the first edit round), where b is the beam width, s is the number of search levels in binary search over the masking percentages, and m is the number of generations sampled for each masking percentage. Our experiments required 180 forward passes, 180 decoded sequences, and 3 backward passes for edit rounds after the first.

While efficient search for targeted edits is an open challenge in other fields of machine learning (Russell, 2019; Dandl et al., 2020), this problem is even more challenging for language data, as the space of possible perturbations is much larger than for tabular data. An important future direction is to develop more efficient methods of finding edits.

This shortcoming prevents us from finding edits that are minimal in a precise sense. In particular, we may be interested in a constrained notion of minimality that defines an edit $e(\boldsymbol{x})$ as minimal if there exists no subset of $e(\boldsymbol{x})$ that results in the contrast prediction. Future work might consider creating methods to produce edits with this property.

6 Related Work

The problem of generating minimal contrastive edits, also called counterfactual explanations (Wachter et al., 2017),[12] has previously been explored for tabular data (Karimi et al., 2020) and images (Hendricks et al., 2018; Goyal et al., 2019; Looveren and Klaise, 2019) but less for language. Recent work explores the use of minimal edits changing true labels for evaluation (Gardner et al., 2020) and data augmentation (Kaushik et al., 2020; Teney et al., 2020), whereas we focus on minimal edits changing model predictions for *explanation*.

Contrastive Explanations within NLP There exist limited methods for automatically generating contrastive explanations of NLP models. Jacovi and Goldberg (2020) define contrastive highlights, which are determined by the inclusion of contrastive features; in contrast, our contrastive edits specify *how* to edit (vs. whether to include) features and can insert new text.[13] Li et al. (2020a) generate counterfactuals using linguistically-informed transformations (LIT), and Yang et al. (2020) generate counterfactuals for binary financial text classification using grammatically plausible single-word edits (REP-SCD). Because both methods rely on manually curated, task-specific rules, they cannot be easily extended to tasks without predefined label spaces, such as RACE.[14] Most recently, Jacovi et al. (2021) propose a method for producing contrastive explanations in the form of latent representations; in contrast, MICE edits are made at the textual level and are therefore more interpretable.

This work also has ties to the literature on causal explanation (Pearl, 2009). Recent work within NLP derives causal explanations of models through counterfactual interventions (Feder et al., 2021; Vig et al., 2020). The focus of our work is the largely unexplored task of creating targeted interventions for language data; however, the question of how to derive causal relationships from such interventions remains an interesting direction for future work.

Counterfactuals Beyond Explanations Concurrent work by Madaan et al. (2021) applies con-

[12] Formally, methods for producing targeted counterfactual explanations solve the same task as MICE. However, not all contrastive explanations are counterfactual explanations; contrastive explanations can take forms beyond contrastive edits, such as free-text rationales (Liang et al., 2020) or highlights (Jacovi and Goldberg, 2020). In this paper, we choose to refer to MICE edits as "contrastive" rather than "counterfactual" because we seek to argue for the utility of *contrastive explanations* of model predictions more broadly; we present MICE as one method for producing contrastive explanations of a particular form and hope future work will explore different forms of contrastive explanations.

[13] See Appendix D for a longer discussion about the advantage of inserting new text in explanations, which MICE edits can do but methods that attribute feature importance (i.e. highlights) cannot.

[14] LIT relies on hand-crafted transformation for NLI tasks based on linguistic knowledge, and REP-SCD makes antonym-based edits using manually curated, domain-specific lexicons for each label.

trolled text generation methods to generate targeted counterfactuals and explores their use as test cases and augmented examples in the context of classification. Another concurrent work by Wu et al. (2021) presents POLYJUICE, a general-purpose, untargeted counterfactual generator. Very recent work by Sha et al. (2021), introduced after the submission of MICE, proposes a method for targeted contrastive editing for Q&A that selects answer-related tokens, masks them, and generates new tokens. Our work differs from these works in our novel framework for efficiently finding *minimal* edits (MICE Stage 2) and our use of edits as explanations.

Connection to Adversarial Examples Adversarial examples are minimally edited inputs that cause models to incorrectly change their predictions despite no change in true label (Jia and Liang, 2017; Ebrahimi et al., 2018; Pal and Tople, 2020). Recent methods for generating adversarial examples also preserve fluency (Zhang et al., 2019; Li et al., 2020b; Song et al., 2020)[15]; however, adversarial examples are designed to find *erroneous* change in model outputs; contrastive edits place no such constraint on model correctness. Thus, current approaches to generating adversarial examples, which can exploit semantics-preserving operations (Ribeiro et al., 2018) such as paraphrasing (Iyyer et al., 2018) or word replacement (Alzantot et al., 2018; Ren et al., 2019; Garg and Ramakrishnan, 2020), cannot be used to generate contrastive edits.

Connection to Style Transfer The goal of style transfer is to generate minimal edits to inputs to result in a target style (sentiment, formality, etc.) (Fu et al., 2018; Li et al., 2018; Goyal et al., 2020). Most existing approaches train an encoder to learn style-agnostic latent representation of inputs and train attribute-specific decoders to generate text reflecting the content of inputs but exhibiting a different target attribute (Fu et al., 2018; Li et al., 2018; Goyal et al., 2020). Recent works by Wu et al. (2019) and Malmi et al. (2020) adopt two-stage approaches that first identify where to make edits and then make them using pretrained language models. Such approaches can only be applied to generate contrastive edits for classification tasks with well-defined "styles," which exclude more complex tasks such as question answering.

[15] Song et al. (2020) propose a method to produce fluent *semantic collisions*, which they call the "inverse" of adversarial examples.

7 Conclusion

We argue that contrastive edits, which change the output of a PREDICTOR to a given contrast prediction, are effective explanations of neural NLP models. We propose MINIMAL CONTRASTIVE EDITING (MICE), a method for generating such edits. We introduce evaluation criteria for contrastive edits that are motivated by human contrastive explanations—minimality and fluency—and show that MICE edits for the IMDB, NEWSGROUPS, and RACE datasets are contrastive, fluent, and minimal. Through qualitative analysis of MICE edits, we show that they have utility for robust and reliable NLP system development.

8 Broader Impact Statement

MICE is intended to aid the interpretation of NLP models. As a model-agnostic explanation method, it has the potential to impact NLP system development across a wide range of models and tasks. In particular, MICE edits can benefit NLP model developers in facilitating debugging and exposing dataset artifacts, as discussed in §4. As a consequence, they can also benefit downstream users of NLP models by facilitating access to less biased and more robust systems.

While the focus of our work is on interpreting NLP models, there are potential misuses of MICE that involve other applications. Firstly, malicious actors might employ MICE to generate adversarial examples; for instance, they may aim to generate hate speech that is minimally edited such that it fools a toxic language classifier. Secondly, naively applying MICE for data augmentation could plausibly lead to less robust and more biased models: Because MICE edits are intended to expose issues in models, straightforwardly using them as additional training examples could reinforce existing artifacts and biases present in data. To mitigate this risk, we encourage researchers exploring data augmentation to carefully think about how to select and label edited instances.

We also encourage researchers to develop more efficient methods of generating minimal contrastive edits. As discussed in §5, a limitation of MICE is its computational demand. Therefore, we recommend that future work focus on creating methods that require less compute.

References

David Alvarez-Melis, Hal Daumé III, Jennifer Wortman Vaughan, and Hanna Wallach. 2019. Weight of evidence as a basis for human-oriented explanations. In *Workshop on Human-Centric Machine Learning at the 33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.*

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani Srivastava, and Kai-Wei Chang. 2018. Generating natural language adversarial examples. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2890–2896, Brussels, Belgium. Association for Computational Linguistics.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. 2019. Interpretable neural predictions with differentiable binary variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy. Association for Computational Linguistics.

Yonatan Belinkov and James Glass. 2019. Analysis Methods in Neural Language Processing: A Survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems*, volume 31, pages 9539–9549. Curran Associates, Inc.

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi Jaakkola. 2019. A game theoretic approach to class-wise selective rationalization. In *Advances in Neural Information Processing Systems*, volume 32, pages 10055–10065. Curran Associates, Inc.

Seth Chin-Parker and Julie A Cantelon. 2017. Contrastive constraints guide explanation-based category learning. *Cognitive Science*, 41 6:1645–1655.

Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. 2020. Multi-objective counterfactual explanations. In *Parallel Problem Solving from Nature – PPSN XVI*, pages 448–469, Cham. Springer International Publishing.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020. ERASER: A benchmark to evaluate rationalized NLP models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4443–4458. Association for Computational Linguistics.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. 2018. HotFlip: White-box adversarial examples for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 31–36, Melbourne, Australia. Association for Computational Linguistics.

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)*, pages 889–898, Melbourne, Australia. Association for Computational Linguistics.

Amir Feder, Nadav Oved, Uri Shalit, and Roi Reichart. 2021. CausaLM: Causal Model Explanation Through Counterfactual Language Models. *Computational Linguistics*, pages 1–54.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *AAAI Conference on Artificial Intelligence*.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181. Association for Computational Linguistics.

Navita Goyal, Balaji Vasan Srinivasan, N. Anandhavelu, and Abhilasha Sancheti. 2020. Multi-dimensional style transfer for partially annotated data using language models as discriminators. *ArXiv*, arXiv:2010.11578.

Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Counterfactual visual explanations. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2376–2384, Long Beach, California, USA. PMLR.

Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. 2018. Grounding visual explanations. In *Computer Vision – ECCV 2018*, pages 269–286, Cham. Springer International Publishing.

Denis Hilton. 2017. *Social Attribution and Explanation*. Oxford University Press.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *International Conference on Learning Representations*.

Mohit Iyyer, John Wieting, Kevin Gimpel, and Luke Zettlemoyer. 2018. Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Alon Jacovi and Y. Goldberg. 2020. Aligning faithful interpretations with their social attribution. *ArXiv*, arXiv:2006.01067.

Alon Jacovi, Swabha Swayamdipta, Shauli Ravfogel, Yanai Elazar, Yejin Choi, and Yoav Goldberg. 2021. Contrastive explanations for model interpretability. ArXiv:2103.01378.

Sarthak Jain, Sarah Wiegreffe, Yuval Pinter, and Byron C. Wallace. 2020. Learning to faithfully rationalize by construction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4459–4473. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

Amir-Hossein Karimi, G. Barthe, B. Balle, and I. Valera. 2020. Model-agnostic counterfactual explanations for consequential decisions. Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS).

Divyansh Kaushik, Eduard Hovy, and Zachary Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data. In *International Conference on Learning Representations*.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117, Austin, Texas. Association for Computational Linguistics.

Chuanrong Li, Lin Shengshuo, Zeyu Liu, Xinyi Wu, Xuhui Zhou, and Shane Steinert-Threlkeld. 2020a. Linguistically-informed transformations (LIT): A method for automatically generating contrast sets. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 126–135, Online. Association for Computational Linguistics.

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)*, pages 1865–1874, New Orleans, Louisiana. Association for Computational Linguistics.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020b. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202. Association for Computational Linguistics.

Weixin Liang, James Zou, and Zhou Yu. 2020. ALICE: Active learning with contrastive natural language explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4380–4391, Online. Association for Computational Linguistics.

Peter Lipton. 1990. Contrastive explanation. *Royal Institute of Philosophy Supplement*, 27:247–266.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*, arXiv:1907.11692.

Arnaud Van Looveren and Janis Klaise. 2019. Interpretable counterfactual explanations guided by prototypes. *ArXiv*, arXiv:1907.02584.

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

Nishtha Madaan, Inkit Padhi, Naveen Panwar, and Diptikalyan Saha. 2021. Generate your counterfactuals: Towards controlled counterfactual generation for text. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Eric Malmi, Aliaksei Severyn, and Sascha Rothe. 2020. Unsupervised text style transfer with padded masked language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8671–8680. Association for Computational Linguistics.

Tim Miller. 2019. Explanation in Artificial Intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. 2020. WT5?! training text-to-text models to explain their predictions. arXiv:2004.14546.

B. Pal and S. Tople. 2020. To transfer or not to transfer: Misclassification attacks against transfer learned text classifiers. *ArXiv*, arXiv:2001.02438.

Judea Pearl. 2009. *Causality: Models, Reasoning and Inference*, 2nd edition. Cambridge University Press, USA.

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):1–67.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

D Raj Reddy. 1977. Speech understanding systems: A summary of results of the five-year research effort.

Shuhuai Ren, Yihe Deng, Kun He, and Wanxiang Che. 2019. Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Mireia Ribera and Àgata Lapedriza. 2019. Can We Do Better Explanations? A Proposal of User-Centered Explainable AI. In *ACM IUI Workshop*.

Chris Russell. 2019. Efficient search for diverse coherent explanations. In *Proceedings of the Conference on Fairness, Accountability, and Transparency*.

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*, pages 2699–2712. Association for Computational Linguistics.

Lei Sha, Patrick Hohenecker, and Thomas Lukasiewicz. 2021. Controlling text edition by changing answers of specific questions. ArXiv:2105.11018.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Workshop Track Proceedings*.

Jacob Sippy, Gagan Bansal, and Daniel S. Weld. 2020. Data staining: A method for comparing faithfulness of explainers. In *2020 ICML Workshop on Human Interpretability in Machine Learning (WHI 2020)*.

D. Smilkov, Nikhil Thorat, Been Kim, F. Viégas, and M. Wattenberg. 2017. Smoothgrad: removing noise by adding noise. In *ICML Workshop on Visualization for Deep Learning*.

Congzheng Song, Alexander Rush, and Vitaly Shmatikov. 2020. Adversarial semantic collisions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4198–4210, Online. Association for Computational Linguistics.

Kaiser Sun and Ana Marasović. 2021. Effective attention sheds light on interpretability. In *Findings of ACL*.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, page 3319–3328. JMLR.org.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Damien Teney, Ehsan Abbasnejad, and A. V. D. Hengel. 2020. Learning what makes a difference from counterfactual examples and gradient supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Bas C Van Fraassen. 1980. *The scientific image*. Oxford University Press.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.

S. Wachter, Brent D. Mittelstadt, and Chris Russell. 2017. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *European Economics: Microeconomics & Industrial Organization eJournal*.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Tongshuang Wu, Marco Túlio Ribeiro, J. Heer, and Daniel S. Weld. 2021. Polyjuice: Automated, general-purpose counterfactual generation. arXiv:2101.00288.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5271–5277. International Joint Conferences on Artificial Intelligence Organization.

Linyi Yang, Eoin Kenny, Tin Lok James Ng, Yi Yang, Barry Smyth, and Ruihai Dong. 2020. Generating plausible counterfactual explanations for deep transformers in financial text classification. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6150–6160, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Mo Yu, Shiyu Chang, Yang Zhang, and Tommi Jaakkola. 2019. Rethinking cooperative rationalization: Introspective extraction and complement control. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4094–4103, Hong Kong, China. Association for Computational Linguistics.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 260–267, Rochester, New York. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From Recognition to Cognition: Visual Commonsense Reasoning. In *CVPR*.

Huangzhao Zhang, Hao Zhou, Ning Miao, and Lei Li. 2019. Generating fluent adversarial examples for natural languages. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5569, Florence, Italy. Association for Computational Linguistics.