Counterfactuals As a Means for Evaluating Faithfulness of Attribution Methods in Autoregressive Language Models

Sepehr Kamahi[1], **Yadollah Yaghoobzadeh**[1,2]
[1]School of Electrical and Computer Engineering
College of Engineering, University of Tehran, Tehran, Iran
[2]Tehran Institute for Advanced Studies, Khatam University, Tehran, Iran
sepehr.kamahi@ut.ac.ir, y.yaghoobzadeh@ut.ac.ir

Abstract

Despite the widespread adoption of autoregressive language models, explainability evaluation research has predominantly focused on span infilling and masked language models (MLMs). Evaluating the faithfulness of an explanation method—how accurately the method explains the inner workings and decision-making of the model—is very challenging because it is very hard to separate the model from its explanation. Most faithfulness evaluation techniques corrupt or remove some input tokens considered important according to a particular attribution (feature importance) method and observe the change in the model's output. This approach creates out-of-distribution inputs for causal language models (CLMs) due to their training objective of next token prediction. In this study, we propose a technique that leverages counterfactual generation to evaluate the faithfulness of attribution methods for autoregressive language modeling scenarios. Our technique creates fluent and in-distribution counterfactuals that makes evaluation protocol more reliable. The code is available at https://github.com/Sepehr-Kamahi/faith.

1 Introduction

Most state-of-the-art NLP systems use autoregressive transformer-based language models (Touvron et al., 2023; Brown et al., 2020; Groeneveld et al., 2024). Since these models are opaque, there is great motivation to understand their decision-making process, and so, explanation methods are becoming increasingly important.

Attribution or feature importance (FI) is a common approach for explaining models (Zhao et al., 2023). Attribution methods' objective is to explain which input features are most salient in the model's predictions. These methods achieve this objective by assigning a scalar to each input feature, representing its significance to the model's decision-making process. Input features in most NLP tasks are sub-words or some combination of them.

A pressing issue for testing and evaluating faithfulness of attribution methods is that most techniques focus on denoising language models or masked language models (Kobayashi et al., 2020, 2021; Ferrando et al., 2022b; Modarressi et al., 2022, 2023; Mohebbi et al., 2023), and most recent work dealing with autoregressive language models has focused on the plausibility of attributions (Yin and Neubig, 2022; Ferrando et al., 2023). Plausibility (or persuasiveness) might be an objective of the explainer, but the explainee's objective is to understand the model's decision-making process, not being persuaded that the decision is correct (Jacovi and Goldberg, 2021).

Almost all previous faithfulness evaluation techniques corrupt the input in one way or another, i.e., masking or erasing important tokens according to the attribution technique, and then looking at the change in the prediction. These techniques probably work fine for MLMs, which are pre-trained for mask and span infilling. However, for autoregressive language models like GPT-2, pre-trained for next token prediction, masking or erasing creates an out-of-distribution (OOD) input for the model. In this case, it is unclear whether corruption techniques evaluate the informativeness of the corrupted tokens or the robustness of the model to unnatural text and the artifacts introduced during test time (Hooker et al., 2019). Furthermore, inputs being OOD makes the explanation socially misaligned Hase et al. (2021); meaning what people expect from FI explanations—demonstrating which features are the most important in the model's decision-making—and what the FI shows become different because FI becomes dependent on model priors rather than removal of features that the model has learned to rely on.

In this work, inspired by counterfactual generation—changing the model's input in a way that flips the output—we develop a technique for evalu-

ating the faithfulness of attribution methods in the autoregressive generation scenario. We use counterfactual generators to change the input, focusing on the important tokens specified by the attribution methods, and make sure that the input to the model is natural, fluent, and in-distribution. This way, we know that the change in the model's prediction is because of changing the important tokens and not because of the input being out of distribution.

We argue that if an attribution method helps the counterfactual generator to change the model's prediction with fewer changes, that method knows more about the model's inner workings, which means it is more faithful. Also, because of the large output space of autoregressive language models like GPT-2 and LLaMA, including often thousands of vocabulary items, looking at the entire output space does not provide much insight. We use contrastive explanation Yin and Neubig (2022), which means looking only at the token predicted by the model and a foil token.

We apply our faithfulness evaluation approach to several attribution techniques including gradient norm, gradient $\times$ input, Erasure, KernelSHAP, and integrated gradient in the next word prediction of two LMs: our fine-tuned Gemma-2b and off-the-shelf Gemma-2b-instruct (Team et al., 2024).

Our contributions are as follows: (i) We introduce a novel faithfulness evaluation protocol, focused on maintaining the input data distribution of the model, suitable for the attribution methods of language models. (ii) We evaluate and rank several popular attribution methods using our approach. (iii) We demonstrate the difference in sensitivity between fine-tuned and off-the-shelf models when dealing with out-of-distribution (OOD) data and provide a solution.

2 Related work

Evaluating Explanations. Current faithfulness metrics mostly use removing important tokens or retraining only on important tokens identified by the attribution methods, (Chan et al., 2022). Abnar and Zuidema (2020) use agreement with gradient and ablation methods as an evaluation of their explanation methods. Wiegreffe and Pinter (2019) acknowledges that gradient methods should not be treated as ideal or ground truth nonetheless use the gradient as a proxy of the model's intrinsic semantics. Explanations' trustworthiness is task- and model-specific (Bastings et al., 2022), and different attribution methods give deeply inconsistent results Neely et al. (2022). hence using one explanation method as the standard or the ground truth in all scenarios does not seem to be justifiable. DeYoung et al. (2020) introduce comprehensiveness, if only the chosen important tokens are used to make the prediction (are highlighted inputs necessary), and sufficiency, if the chosen important tokens on their own are sufficient to make the prediction. Carton et al. (2020) introduce a normalized version of comprehensiveness and sufficiency by dividing these measures into null difference. The Null difference is the sufficiency of an empty input or comprehensiveness of a full input. It is unclear whether corruption techniques evaluate the informativeness of the corrupted tokens or the robustness of the model to unnatural text and the artifacts introduced during test time.

Han et al. (2020) and Jain et al. (2020) consider an attribution method to be either faithful or unfaithful, with no middle ground, and use the term 'faithful by construction.' In contrast, other works argue that faithfulness exists on a spectrum and that we should evaluate the 'degree of faithfulness' of an explanation method (Jacovi and Goldberg, 2020). We adopt the latter approach, searching for a sufficiently faithful explanation method for our tasks.

Atanasova et al. (2023) use counterfactuals to evaluate faithfulness of natural language explanations—When we want the model to tell us in natural language why it made a particular decision—they use counterfactual generation techniques from (Ross et al., 2021).

Another line of work tries to evaluate explanations using uncertainty estimation. Slack et al. (2021) develop a Bayesian framework for generating feature importance estimates along with their associated uncertainty in the form of credible intervals.

OOD Problem in Explainability. Hooker et al. (2019) and Vafa et al. (2021) suggest retraining or fine-tuning the model with inputs that are partly erased to achieve a model that has the same train and evaluation distribution. However, this retraining is expensive and not always possible. Kim et al. (2020) try to solve the OOD problem by making the explanation itself in-distribution.

Feature Importance (Attribution). attributions are local explanations that assign a score to each input feature (token embeddings in most NLP tasks). The score represents how important that feature is

for the predictor model according to that explanation method. We can categorize these methods into four types. i) Perturbation-based methods which work by perturbing the input examples such as removing or masking to measure feature importance (Li et al., 2016, 2017; Feng et al., 2018; Wu et al., 2020). ii) Gradient-Based methods determine the importance of each input feature by measuring the derivative of output with respect to each input (Mohebbi et al., 2021; Kindermans et al., 2019; Sundararajan et al., 2017; Lundstrom et al., 2022; Enguehard, 2023; Sanyal and Ren, 2021; Sikdar et al., 2021). iii) Surrogate-based methods use a simple, interpretable model to explain the original complex, black-box model (Ribeiro et al., 2016; Lundberg and Lee, 2017; Kokalj et al., 2021). iv) Decomposition-based methods try to break down the importance score into linear contributions from the input (Montavon et al., 2019; Voita et al., 2021; Chefer et al., 2021; Modarressi et al., 2022; Ferrando et al., 2022a).

3 Our method

Our faithfulness evaluation protocol consists of two models: the counterfactual generator model and the predictor model. We aim to evaluate the faithfulness of attribution methods for the predictor model. Due to the large output space of LMs, we use contrastive explanations proposed by Yin and Neubig (2022) that measure the attribution of input tokens for a contrastive model decision. Contrastive attributions try to identify the most important tokens that led the model to predict the target y_t instead of a foil y_f. Then we use a separate editor model to change these important tokens to generate counterfactuals, i.e. generating examples that will make the original predictor model more likely to predict the foil.

The protocol to evaluate attributions consists of two phases. The first phase involves creating the editor that can generate counterfactuals. In the second phase of evaluating attributions, we use the editor and predictor together to determine what percentage of tokens the editor needs to change in order to flip the predictor model's prediction. Figure 2 demonstrates the seconds phase.

Our approach for creating the editor is fine-tuning an autoregressive language model specifically for counterfactual generation. For fine-tuning, we add two tokens to the embedding space and the tokenizer, specifically '<mask>' and '<counterfactual>'. For creating training examples for our counterfactual generator, inspired by Wu et al. (2021) and Donahue et al. (2020), we randomly mask between 5 and 50 percent of the tokens, then append each example's label to it, e.g. positive or negative for SST-2 dataset, then append the <counterfactual>token, and lastly, we append the original unmasked example. The training example creation is shown in Figure 3.

```
<masked text> <foil label><counterfactual>
```

Figure 1: Prompting techniques used for counterfactual generation in the second phase.

In the second phase of evaluating attributions, First, we give a sentence to the predictor, then use an attribution method that was applied to the predictor to identify the most important tokens for the predictor's decision-making process. We begin by replacing 10% of these most important tokens with '<mask>' and present the masked sentence and the foil label (The label with the second highest logit) to the *editor* to generate a counterfactual sentence (one that flips the prediction of the predictor model). If unsuccessful in flipping the prediction, we incrementally increase masking by 10% until we either flip the prediction or reach a masking threshold of 50%. This evaluation protocol is shown in Figure 2. The prompting technique used for counterfactual generation during this second phase is shown in Figure 1. The attribution technique that helps us identify the most critical tokens for creating counterfactuals and helps creating counterfactuals with the least amount of change in the original text, is the one that provides the most faithful representation of the predictor's decision-making process.

4 Experimental Setup

4.1 Datasets

Three datasets are used for evaluating faithfulness. SST-2 (Socher et al., 2013) and IMDB (Maas et al., 2011) which are binary classification, and AG-News (Zhang et al., 2015) a four class classification dataset.

Faithfulness evaluation datasets should not have gold attribution labels because we do not want human intuition in faithfulness evaluation. We want to know how the model makes a prediction (Jacovi

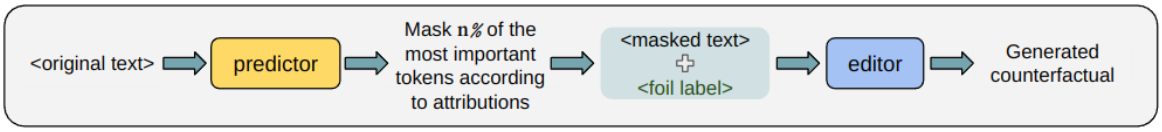

Figure 2: Our process of generating counterfactuals for evaluating attribution methods. The predictor (an LM), generates a label for the given text, and an attribution method specifics the most important tokens. We mask the top n% of them and ask an editor (another LM) to change the label of the input text by filling in the masked tokens. If the attribution method is more faithful, then the needed n% should be a lower number.

```
<randomly masked text> <original label><counterfactual><unmasked text>
Example:
<mask> a clear<mask> of<mask> positive<counterfactual> a clear sense of purpose
```

Figure 3: Training example creation for fine-tuning the counterfactual generator, and one given sample.

and Goldberg, 2020).

4.2 Models

4.2.1 Editor Models

For the editor model our method is very similar to Wu et al. (2021) which uses GPT-2 which is a decoder-only causal model for generating counterfactuals. We use three more modern decoder-only models. GPT-J-6B Wang and Komatsuzaki (2021) which we call gptj , and two sizes of Pythia. Pythia-1.4B (pythia1), and Pythia-2.8B (pythia2) (Biderman et al., 2023). We fine-tune these models in accordance with Section 3. pythia1 model is fully fine-tuned but the other two are tuned using Low Rank Adaptation (LoRA) (Hu et al., 2022). We train these models for 8 epochs using dynamic masking (Liu et al., 2019), which means masking each example differently in each epoch.

4.2.2 Predictor Models

We use Gemma-2b Team et al. (2024) as the predictor. We fine-tune the raw language model for the three tasks (gemma-ft). We use LoRA for fine-tuning. We also use an off-the-shelf instruct-tuned version (gemma-it) in zero-shot scenario. We compare these two versions (fine-tuned and non-fine-tuned) extensively for attribution evaluation.

4.3 Attribution Methods

Here we detail the six attribution methods that we use which are all widely used methods. We use all attribution methods in a contrastive way (Yin and Neubig, 2022). Contrastive attributions measure which features from the input make the foil token y_f more likely and the target token y_t less likely. We denote contrastive, target, and foil attributions by S^C, S^t, and S^f respectively:

$$S^C = S^t - S^f \tag{1}$$

We use implementation of these attribution methods provided by Yin and Neubig (2022) (for Gradient × input, gradient norm and erasure) and by Captum (Miglani et al., 2023) (for KernelSHAP and Integrated Gradient).

4.3.1 Gradient Norm

We can calculate attributions based on the norm of the gradient of the model prediction, with respect to the input (Simonyan et al., 2013; Li et al., 2016). Gradient with respect to feature x_i:

$$g(x_i) = \nabla_{x_i} q(y_t|x)$$

Where $q(y_t|x)$ is the model output for token y_t given the input x. The contrastive gradient:

$$g^C(x_i) = \nabla_{x_i} \left(q(y_t|\boldsymbol{x}) - q(y_f|\boldsymbol{x})\right)$$

We will use both norm one and norm two:

$$S^C_{GN1}(x_i) = ||g^C(x_i)||_{L1}$$

$$S^C_{GN2}(x_i) = ||g^C(x_i)||_{L2}$$

4.3.2 Gradient × Input

In gradient × input (gradinp) method (Shrikumar et al., 2016; Denil et al., 2014), we take the dot product of the gradient with the input token embedding x_i:

$$S_{GI}(x_i) = g(x_i) \cdot x_i$$

By multiplying the gradient with the input embedding, we also account for how much each token is expressed in the attribution score. The Contrastive Gradient $\times$ Input is:

$$S^C_{GI}(x_i) = g^C(x_i) \cdot x_i$$

4.3.3 Erasure

Erasure-based methods measure the importance of each token by erasing it and seeing the effect on the model output (Li et al., 2017). This is achieved by taking the difference of model output with the full input x and part of the input zeroed out, $x_{\neg i}$:

$$S^t_E(x_i) = q(y_t|x) - q(y_t|x_{\neg i})$$

For the contrastive case:

$$S^C_E = (q(y_t|x) - q(y_t|x_{\neg i})) - (q(y_f|x) - q(y_f|x_{\neg i}))$$

4.3.4 KernelSHAP

KernelSHAP (Lundberg and Lee, 2017) explains the prediction of a classifier q by learning a linear model ϕ locally around each prediction. The objective function of KernelSHAP constructs an explanation that approximates the behavior of q accurately in the neighborhood of x. More important features have higher weights in this linear model ϕ. Let Z be a set of N randomly sampled perturbations around x:

$$S^t_\phi = \arg\min_\phi \sum_{z \in Z} [q(y_t|z) - \phi^T z]^2 \pi_x(z) \quad (2)$$

KernelSHAP uses a kernel π_x that satisfies certain principles when input features are considered agents of a cooperative game in game theory. We use equation 2 in a contrastive way. First we normalize S^t_ϕ and S^f_ϕ by dividing to their $L2$ norm and then subtracting:

$$S^C_\phi = \frac{S^t_\phi}{||S^t_\phi||} - \frac{S^f_\phi}{||S^f_\phi||} \quad (3)$$

4.3.5 Integrated Gradients

Integrated Gradients (IG) (Sundararajan et al., 2017) is a gradient-based method which addresses the problem of saturation: gradients may get close to zero for a well-fitted function. IG requires a baseline $\mathbf{b}$ as a way of contrasting the given input with information being absent. For input i, we compute:

$$S^t_{IG} = \frac{1}{m} \sum_{k=1}^{m} \nabla_{x_i} q\left(y_t \middle| b + \frac{k}{m}(x - b)\right) \cdot (x_i - b_i) \quad (4)$$

That is, we average over m gradients, with the inputs to f_t being linearly interpolated between the baseline and the original input x in m steps. We then take the dot product of that averaged gradient with the input embedding $\mathbf{x}_i$ minus the baseline.

We use zero vector baseline (Mudrakarta et al., 2018), and 5 steps. The contrastive case becomes:

$$S^C_{IG} = \frac{S^t_{IG}}{||S^t_{IG}||} - \frac{S^f_{IG}}{||S^f_{IG}||} \quad (5)$$

5 Results and Discussion

5.1 The Out-of-Distribution problem

Why should we use counterfactuals instead of erasing the important tokens or replacing important tokens with unimportant tokens? First, we show that our counterfactual generators generate in-distribution text for the predictor. Second, we show that the rankings of faithfulness of attribution methods are consistent with each other when we are using a counterfactual generator for token replacement but these rankings differ when other replacement methods are used.

To achieve our first goal, which is to show that the generated counterfactuals are in-distribution, we use an OOD detection technique to measure what percentage of our generated inputs are OOD. Prominent methods to detect OOD use a threshold and consider anything higher than that threshold as OOD (Chen et al., 2023). To calculate the threshold, for each dataset, we measure the negative log-likelihood (NLL) of the 200 original examples using different predictors—fine-tuned and instruct-tuned—and consider the 99th percentile of the NLLs as the threshold for being OOD. The reason we use NLL to detect OOD is that the type of shift we want to detect is background shift. We can classify the types of OOD data as either semantic or background shift (Arora et al., 2021). Semantic features have a strong correlation with the label, and semantic shift happens when we encounter unseen classes at test time, while background features consist of population-level statistics that do not depend on the label and focus on the style of the text.

In the evaluation of faithfulness by corrupting the input, we are not introducing new labels or classes. We are changing the style of text; therefore, we want to detect background shift. There are two common types of OOD detection methods: calibration and density estimation. Density estimation methods, e.g., perplexity (PPL), outperform cali-

Editor	gradnorm1		Erasure		KernelSHAP	
	gemma-ft	gemma-it	gemma-ft	gemma-it	gemma-ft	gemma-it
pythia1 (ours)	1.1	1.4	1.3	1.6	0.7	1.7
pythia2 (ours)	0.4	2.6	0.9	1.3	0.8	2.3
gptj (ours)	0.7	8.3	2.0	10.9	0.9	6.4
erase	0.3	19.9	2.3	32.8	0.6	81.4
unk	0.6	97.5	1.8	97.3	1.3	99.8
mask	0.0	94.8	0.5	93.3	0.0	98.5
att-zero	0.1	80.9	0.1	62.6	0	74.1

Table 1: OOD percentage when our counterfactual editor models generate samples, compared to other replacement methods (erase, unk, mask, and att-zero methods). This represents the percentage of corrupted examples that fall outside the 99th percentile of the NLL of the original sentences in the SST-2 dataset (lower is better). Scenarios with very high OOD percentages are highlighted.

Attribution method	SST-2			IMDB			AG-News		
	pythia1	pythia2	gptj	pythia1	pythia2	gptj	pythia1	pythia2	gptj
gradnorm1	33.5	**34.8**	**32.2**	**29.1**	**30.3**	**32.4**	42.9	45.1	44.0
gradnorm2	**33.4**	35.6	32.6	31.0	30.5	**32.4**	42.6	44.4	43.9
gradinp	40.5	41.8	40.8	36.1	36.3	36.5	43.1	44.6	**42.2**
erasure	35.5	36.6	33.4	32.7	32.7	34.4	**42.0**	**42.7**	43.0
IG	45.7	45.8	43.7	43.3	44.3	42.5	43.8	46.7	44.0
KernelSHAP	44.1	45.9	44.9	44.0	43.3	44.2	44.0	46.5	44.3
Random	44.6	46.0	44.3	43.8	42.7	43.2	44.0	46.0	44.0

Table 2: The mean percentage needed to mask to achieve flipping Gemma-ft's label or reaching 50 percent masking in 200 examples of evaluation split in SST-2, IMDB, and AG-News datasets (lower is better). pythia1, pythia2, and gptj models are used to fill the masks and generate counterfactuals.

bration methods under background shifts, while the opposite is true under semantic shift. We will use NLL, which is a very similar notion to PPL. An attribution method shows us which tokens are important, and we will replace those tokens in four ways: (i) using an editor to replace the token (our method), (ii) using a token that is considered unimportant semantically (<unk> token and <mask> token), (iii) erasing the tokens, and (iv) zeroing out the attention mask for important tokens without altering the text itself (att-zero). The baselines (ii) through (iv) are similar to previous work (Hase et al., 2021). In Table 1, it is shown that for both fine-tuned and instruct-tuned predictors, the generated counterfactuals are mostly in-distribution. In Table 1, we show the results for the SST-2 dataset and three attribution methods; other attribution methods and datasets are shown in Appendix A. Each number in the table represents the average of five amounts of replacement (10% to 50%) and 200 examples from evaluation sets.

Chen et al. (2023) demonstrate that fine-tuning renders a model insensitive to non-semantic shift. Their research indicates that fine-tuning eliminates pre-trained task-agnostic knowledge about general linguistic properties, which are crucial for detecting non-semantic shift. Our findings align with these observations. When a predictor is fine-tuned for a specific classification task, such as sentiment analysis on the SST-2 dataset, it is optimized to assign the highest probability to the positive token for examples with positive sentiment. Consequently, this fine-tuned model becomes less sensitive to input corruptions. In our experiment, regardless of the replacement method employed, the resulting input tends to remain in-distribution for the fine-tuned predictors. As evidenced in Table 1, under the Gemma-ft columns, the percentage of out-of-distribution (OOD) examples approaches zero. In contrast, Gemma-it, an off-the-shelf model which is not optimized for a specific dataset, exhibits different behavior. When subjected to various in-

Attribution method	SST-2			IMDB			AG-News		
	pythia1	pythia2	gptj	pythia1	pythia2	gptj	pythia1	pythia2	gptj
gradnorm1	41.4	42.0	**40.3**	42.1	42.3	44.0	46.6	46.8	40.0
gradnorm2	41.5	42.2	40.6	42.3	42.6	43.6	46.6	46.9	39.9
gradinp	42.9	43.4	43.2	**41.2**	**41.4**	**42.2**	45.8	45.3	39.2
erasure	**40.8**	**41.5**	41.0	43.1	42.8	43.8	45.0	45.5	39.5
IG	44.7	44.0	45.4	43.4	43.2	44.0	45.7	45.3	38.6
KernelSHAP	43.6	43.5	44.0	43.7	42.9	43.9	46.1	45.3	**37.2**
Random	44.8	44.7	44.3	45.4	46.1	45.8	**44.9**	**45.2**	39.2

Table 3: The mean percentage needed to mask to achieve flipping Gemma-it's label or reaching 50 percent masking in 200 examples of evaluation split in SST-2, IMDB, and AG-News datasets (lower is better). pythia1, pythia2, and gptj models are used to fill the masks and generate counterfactuals.

put modifications—such as replacing important tokens with semantically neutral ones (e.g., <unk> or <mask> tokens), completely removing tokens, or zeroing out the attention mask for important tokens without altering the text itself—the Gemma-it predictor frequently categorizes these modified inputs as OOD. This disparity in behavior between fine-tuned and off-the-shelf models underscores the impact of task-specific optimization on a model's sensitivity to input perturbations. Notably, the examples remain in-distribution when modified by the counterfactual generator, even for the instruct-tuned predictor. This observation demonstrates that when we don't want to change the predictor model and prefer to use an off-the-shelf model as our predictor, using a counterfactual generator is helpful in evaluating the faithfulness of attribution methods.

To achieve our second goal, which is to demonstrate the consistency of the faithfulness rankings of attribution methods regardless of the counterfactual generator used, and the lack of consistency when another replacement method is applied, we use Spearman's rank correlation, which has been used in previous works (Rong et al., 2022).

We rank the attribution methods' mask percentage needed to flip the label for each example for all seven replacement methods (our three editors, Erase, <unk>, <mask>, and att-zero), get the correlation of these ranks with each other, and then take the average over 200 examples. We show this for the SST-2 dataset in Figure 4. Other datasets yield similar results and are shown in Appendix B. In the top correlation matrix of Figure 4, these average correlations are shown for the fine-tuned predictor. For the fine-tuned predictor, all replacement methods have high average correlation with each other. The middle matrix in Figure 4 shows these correlations when the predictor models are off-the-shelf instruct-tuned models. For the off-the-shelf predictor, only when a counterfactual generator is used do the ranks have high correlation with each other; other replacement methods have low correlation with counterfactual generators. This is likely because when using an instruct-tuned predictor, replacement methods other than counterfactual generators create OOD inputs. The figure shows the rank of explanations using editors has a higher correlation with unk/erase when the predictor model is fine-tuned. The bottom matrix of 4 is the difference between the first and second matrices. It indicates that the correlation difference between fine-tuned and instruct-tuned predictors is near zero when using editors as the replacement method, but the difference with other corruption methods (unk/erase/mask/att-zero) is significant. This suggests that when evaluating explanations on an off-the-shelf instruct-tuned model, it is crucial not to use corrupted OOD text.

5.2 Analysis of FI methods

In Tables 2 and 3, we show the average masking percent needed (the average percentage of tokens the counterfactual generator should change) to flip the label for fine-tuned and instruct-tuned predictor models, respectively. The masking percent is highly correlated with flip-rate—what percentage of labels each counterfactual generator is able to flip by changing the corrupted tokens. In Appendix C, we show flip-rate for both fine-tuned and instruct-tuned predictor models. Attribution methods that are able to flip the labels with less mask percent (i.e. less change) are also able to flip more labels.

For the fine-tuned predictor (Table 2), gradient norm methods consistently outperform others on the SST-2 and IMDB datasets. In contrast, for AG-News, the Erasure method consistently performs the best or near the best. Our results indicate that straightforward methods, such as gradnorm1, gradnorm2, and Erasure, consistently deliver superior performance regardless of the editor used.

For the instruct-tuned predictor (Table 3), the Erasure method yields the best results for the SST-2 dataset, while gradinp demonstrates the best performance on the IMDB dataset. However, no attribution method consistently outperforms random selection for the AG-News dataset. Overall, these findings suggest that attribution methods are less effective when the model is not fine-tuned for the specific task, indicating the need for cautious application of these methods to pretrained and instruct-tuned language models.

6 Conclusion

In this work, we designed a faithfulness evaluation protocol based on counterfactual generation. We demonstrated that the efficacy of attribution methods varies between models fine-tuned on our specific dataset and off-the-shelf instruct-tuned models. We showed that counterfactual generators are effective for evaluating feature attribution because they can produce mostly in-distribution text for the predictor model. This approach allows us to separate the evaluation of the model from the evaluation of the attribution method, as the examples used are mostly in-distribution. We also found high consistency between different counterfactual generators and a lack of consistency with other replacement methods, highlighting the importance of being in-distribution, particularly when evaluating attributions on off-the-shelf models. Finally, we used our protocol to compare different attribution methods.

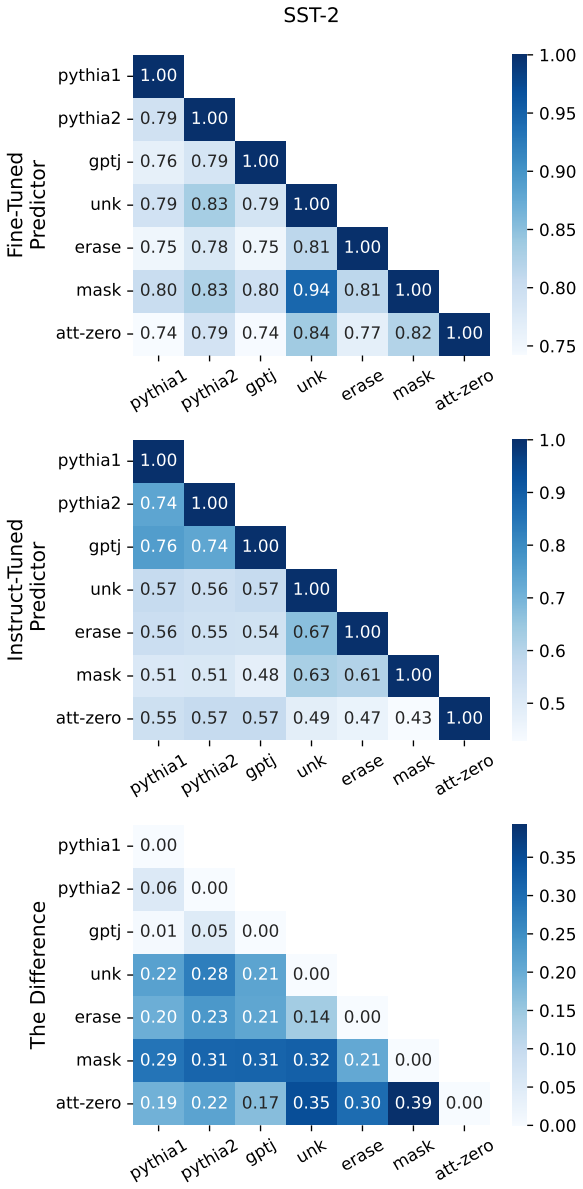

Figure 4: In the top matrix, the average correlation of attribution ranks for the fine-tuned predictor is presented. The middle matrix shows the average correlation of attribution ranks when the predictor is an off-the-shelf model. The bottom matrix displays the difference between the first two, indicating that when using editors, the difference in correlation between the two types of predictors is near zero. However, using other replacement methods results in inconsistent correlations across the two scenarios, likely because these methods produce OOD text for the instruct-tuned model.

7 Limitations

Our work is limited in several aspects: First, we rely on generating counterfactuals, for which a strong generative model is needed. Generating counterfactuals, especially for long sequences, is computationally expensive. Second, the counterfactual generator may unintentionally know the artifacts and shortcuts used by the predictor to flip the label, and this could limit the intended application of our approach. Third, we evaluated our protocol for classification datasets; evaluating it for other tasks like translation is left for future work.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Udit Arora, William Huang, and He He. 2021. Types of out-of-distribution texts and how to detect them. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10687–10701, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pepa Atanasova, Oana-Maria Camburu, Christina Lioma, Thomas Lukasiewicz, Jakob Grue Simonsen, and Isabelle Augenstein. 2023. Faithfulness tests for natural language explanations. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 283–294, Toronto, Canada. Association for Computational Linguistics.

Jasmijn Bastings, Sebastian Ebert, Polina Zablotskaia, Anders Sandholm, and Katja Filippova. 2022. "will you find these shortcuts?" a protocol for evaluating the faithfulness of input salience methods for text classification. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 976–991, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

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*, volume 33, pages 1877–1901. Curran Associates, Inc.

Samuel Carton, Anirudh Rathore, and Chenhao Tan. 2020. Evaluating and characterizing human rationales. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9294–9307, Online. Association for Computational Linguistics.

Chun Sik Chan, Huanqi Kong, and Liang Guanqing. 2022. A comparative study of faithfulness metrics for model interpretability methods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5029–5038, Dublin, Ireland. Association for Computational Linguistics.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer interpretability beyond attention visualization. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 782–791.

Sishuo Chen, Wenkai Yang, Xiaohan Bi, and Xu Sun. 2023. Fine-tuning deteriorates general textual out-of-distribution detection by distorting task-agnostic features. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 564–579, Dubrovnik, Croatia. Association for Computational Linguistics.

Misha Denil, Alban Demiraj, and Nando De Freitas. 2014. Extraction of salient sentences from labelled documents. *arXiv preprint arXiv:1412.6815*.

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, Online. Association for Computational Linguistics.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Joseph Enguehard. 2023. Sequential integrated gradients: a simple but effective method for explaining language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7555–7565, Toronto, Canada. Association for Computational Linguistics.

Shi Feng, Eric Wallace, Alvin Grissom II, Mohit Iyyer, Pedro Rodriguez, and Jordan Boyd-Graber. 2018. Pathologies of neural models make interpretations

difficult. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3719–3728, Brussels, Belgium. Association for Computational Linguistics.

Javier Ferrando, Gerard I. Gállego, Belen Alastruey, Carlos Escolano, and Marta R. Costa-jussà. 2022a. Towards opening the black box of neural machine translation: Source and target interpretations of the transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8756–8769, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Javier Ferrando, Gerard I. Gállego, and Marta R. Costa-jussà. 2022b. Measuring the mixing of contextual information in the transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8698–8714, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Javier Ferrando, Gerard I. Gállego, Ioannis Tsiamas, and Marta R. Costa-jussà. 2023. Explaining how transformers use context to build predictions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5486–5513, Toronto, Canada. Association for Computational Linguistics.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Raghavi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, Tushar Khot, William Merrill, Jacob Morrison, Niklas Muennighoff, Aakanksha Naik, Crystal Nam, Matthew E. Peters, Valentina Pyatkin, Abhilasha Ravichander, Dustin Schwenk, Saurabh Shah, Will Smith, Emma Strubell, Nishant Subramani, Mitchell Wortsman, Pradeep Dasigi, Nathan Lambert, Kyle Richardson, Luke Zettlemoyer, Jesse Dodge, Kyle Lo, Luca Soldaini, Noah A. Smith, and Hannaneh Hajishirzi. 2024. Olmo: Accelerating the science of language models. *Preprint*, arXiv:2402.00838.

Xiaochuang Han, Byron C. Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563, Online. Association for Computational Linguistics.

Peter Hase, Harry Xie, and Mohit Bansal. 2021. The out-of-distribution problem in explainability and search methods for feature importance explanations. In *Advances in Neural Information Processing Systems*, volume 34, pages 3650–3666. Curran Associates, Inc.

Sara Hooker, Dumitru Erhan, Pieter-Jan Kindermans, and Been Kim. 2019. A benchmark for interpretability methods in deep neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 9737–9748. Curran Associates, Inc.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4198–4205, Online. Association for Computational Linguistics.

Alon Jacovi and Yoav Goldberg. 2021. Aligning faithful interpretations with their social attribution. *Transactions of the Association for Computational Linguistics*, 9:294–310.

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, Online. Association for Computational Linguistics.

Siwon Kim, Jihun Yi, Eunji Kim, and Sungroh Yoon. 2020. Interpretation of NLP models through input marginalization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3154–3167, Online. Association for Computational Linguistics.

Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T. Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2019. *The (Un)reliability of Saliency Methods*, pages 267–280. Springer International Publishing, Cham.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2020. Attention is not only a weight: Analyzing transformers with vector norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7057–7075, Online. Association for Computational Linguistics.

Goro Kobayashi, Tatsuki Kuribayashi, Sho Yokoi, and Kentaro Inui. 2021. Incorporating Residual and Normalization Layers into Analysis of Masked Language Models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4547–4568, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Enja Kokalj, Blaž Škrlj, Nada Lavrač, Senja Pollak, and Marko Robnik-Šikonja. 2021. BERT meets shapley: Extending SHAP explanations to transformer-based classifiers. In *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, pages 16–21, Online. Association for Computational Linguistics.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in NLP. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691, San Diego, California. Association for Computational Linguistics.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2017. Understanding neural networks through representation erasure. *Preprint*, arXiv:1612.08220.

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

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Daniel D Lundstrom, Tianjian Huang, and Meisam Razaviyayn. 2022. A rigorous study of integrated gradients method and extensions to internal neuron attributions. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 14485–14508. PMLR.

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.

Vivek Miglani, Aobo Yang, Aram Markosyan, Diego Garcia-Olano, and Narine Kokhlikyan. 2023. Using captum to explain generative language models. In *Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pages 165–173, Singapore. Association for Computational Linguistics.

Ali Modarressi, Mohsen Fayyaz, Ehsan Aghazadeh, Yadollah Yaghoobzadeh, and Mohammad Taher Pilehvar. 2023. DecompX: Explaining transformers decisions by propagating token decomposition. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2649–2664, Toronto, Canada. Association for Computational Linguistics.

Ali Modarressi, Mohsen Fayyaz, Yadollah Yaghoobzadeh, and Mohammad Taher Pilehvar. 2022. GlobEnc: Quantifying global token attribution by incorporating the whole encoder layer in transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 258–271, Seattle, United States. Association for Computational Linguistics.

Hosein Mohebbi, Ali Modarressi, and Mohammad Taher Pilehvar. 2021. Exploring the role of BERT token representations to explain sentence probing results. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 792–806, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hosein Mohebbi, Willem Zuidema, Grzegorz Chrupała, and Afra Alishahi. 2023. Quantifying context mixing in transformers. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3378–3400, Dubrovnik, Croatia. Association for Computational Linguistics.

Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. 2019. *Layer-Wise Relevance Propagation: An Overview*, pages 193–209. Springer International Publishing, Cham.

Pramod Kaushik Mudrakarta, Ankur Taly, Mukund Sundararajan, and Kedar Dhamdhere. 2018. Did the model understand the question? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1896–1906, Melbourne, Australia. Association for Computational Linguistics.

Michael Neely, Stefan F. Schouten, Maurits Bleeker, and Ana Lucic. 2022. A song of (dis)agreement: Evaluating the evaluation of explainable artificial intelligence in natural language processing. *Preprint*, arXiv:2205.04559.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 97–101, San Diego, California. Association for Computational Linguistics.

Yao Rong, Tobias Leemann, Vadim Borisov, Gjergji Kasneci, and Enkelejda Kasneci. 2022. A consistent and efficient evaluation strategy for attribution methods. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 18770–18795. PMLR.

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*, pages 3840–3852, Online. Association for Computational Linguistics.

Soumya Sanyal and Xiang Ren. 2021. Discretized integrated gradients for explaining language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages

10285–10299, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Avanti Shrikumar, Peyton Greenside, Anna Shcherbina, and Anshul Kundaje. 2016. Not just a black box: Learning important features through propagating activation differences. *arXiv preprint arXiv:1605.01713*.

Sandipan Sikdar, Parantapa Bhattacharya, and Kieran Heese. 2021. Integrated directional gradients: Feature interaction attribution for neural NLP 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)*, pages 865–878, Online. Association for Computational Linguistics.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Dylan Slack, Sophie Hilgard, Sameer Singh, and Himabindu Lakkaraju. 2021. Reliable Post hoc Explanations Modeling Uncertainty in Explainability. In *Neural Information Processing Systems (NeurIPS)*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

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 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Keyon Vafa, Yuntian Deng, David Blei, and Alexander Rush. 2021. Rationales for sequential predictions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10314–10332, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation. 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)*, pages 1126–1140, Online. Association for Computational Linguistics.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

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.

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)*, pages 6707–6723, Online. Association for Computational Linguistics.

Zhiyong Wu, Yun Chen, Ben Kao, and Qun Liu. 2020. Perturbed masking: Parameter-free probing for analyzing and interpreting BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4166–4176, Online. Association for Computational Linguistics.

Kayo Yin and Graham Neubig. 2022. Interpreting language models with contrastive explanations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 184–198, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2023. Explainability for large language models: A survey. *Preprint*, arXiv:2309.01029.

A

Table 4 is the OOD percentages for other attribution methods in SST-2 dataset that were not in Table 1. Tables 5 and 6 show OOD percentages in AG-News dataset.

B

Figures 5 and 6 show the difference of correlations in IMDB and AG-News datasets respectively.

C

Tables 7 and 8 show flip-rates for fine-tuned and instruct-tuned datasets respectively.

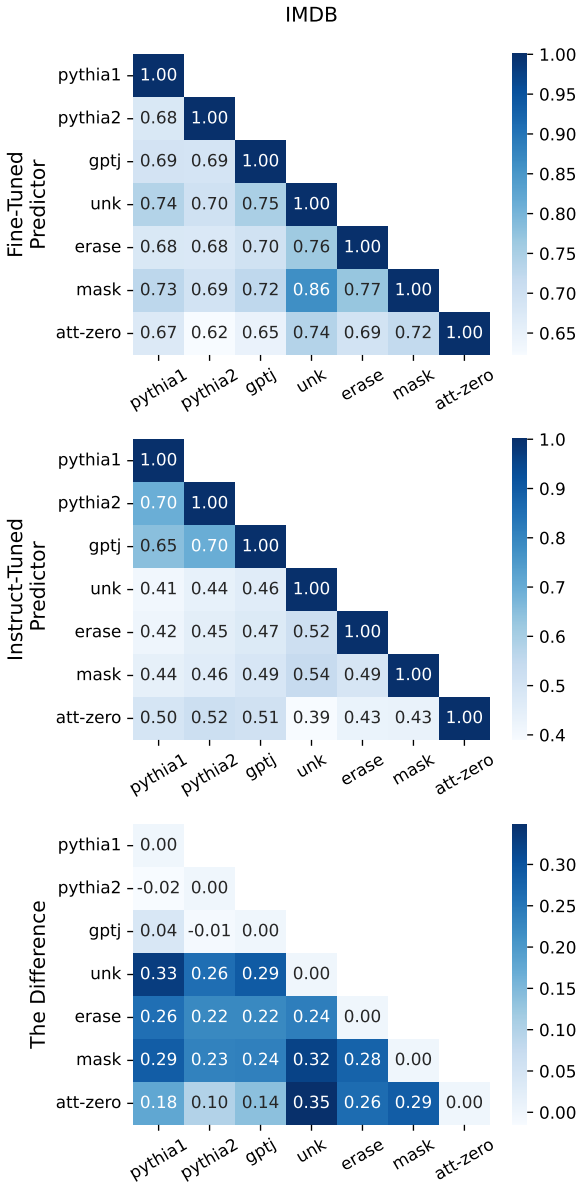

Figure 5: The difference

Editor	gradnorm2		gradinp		integrated gradient	
	gemma-ft	gemma-it	gemma-ft	gemma-it	gemma-ft	gemma-it
pythia1 (ours)	1.4	1.2	1.3	1.5	1.1	0.7
pythia2 (ours)	0.4	2.6	0.8	1.6	1.0	0.8
gptj (ours)	0.8	8.2	2.4	5.2	0.4	8.7
erase	0.5	21.9	1.1	83.8	1.4	77.4
unk	0.6	98.4	1.6	100.0	3.4	99.3
mask	0.0	95.8	0.2	99.5	0.4	97.8
att-zero	0.1	80.8	0.0	70.6	0	65.8

Table 4: OOD percentage when our counterfactual editor models generate samples, compared to other replacement methods (erase, unk, mask, and att-zero methods). This is the percentage of corrupted examples that are out of the 99th percentile of the NLL of the original sentences in SST-2 dataset (lower is better). The scenarios with very high numbers of OODs are highlighted.

Editor	gradnorm1		Erasure		KernelSHAP	
	gemma-ft	gemma-it	gemma-ft	gemma-it	gemma-ft	gemma-it
pythia1 (ours)	2.4	5.0	4.1	4.9	1.6	4.5
pythia2 (ours)	2.4	5.2	3.1	4.9	2.0	6.9
gptj (ours)	1.0	5.0	1.5	4.9	0.9	7.8
erase	3.5	11.5	8.8	46.3	2.3	74.9
unk	1.0	98.2	3.8	99.7	1.3	99.9
mask	0.7	85.7	5.2	96.7	0.5	98.2
att-zero	4.9	79.1	3.2	57.4	0.8	62.6

Table 5: OOD percentage when our counterfactual editor models generate samples, compared to other replacement methods (erase, unk, mask, and att-zero methods). This is the percentage of corrupted examples that are out of the 99th percentile of the NLL of the original sentences in AG-News dataset (lower is better). The scenarios with very high numbers of OODs are highlighted.

Editor	gradnorm2		gradinp		integrated gradient	
	gemma-ft	gemma-it	gemma-ft	gemma-it	gemma-ft	gemma-it
pythia1 (ours)	2.2	5.0	1.4	4.5	1.6	4.5
pythia2 (ours)	2.4	5.2	2.3	5.5	1.3	5.9
gptj (ours)	1.2	5.0	2.0	5.1	1.2	6.4
erase	3.3	11.6	2.6	60.3	1.9	55.5
unk	0.9	98.0	1.7	99.3	1.3	99.9
mask	0.8	86.4	1.5	94.1	0.5	98.2
att-zero	5.1	78.9	1.9	58.7	0.6	50.8

Table 6: OOD percentage when our counterfactual editor models generate samples, compared to other replacement methods (erase, unk, mask, and att-zero methods). This is the percentage of corrupted examples that are out of the 99th percentile of the NLL of the original sentences in AG-News dataset (lower is better). The scenarios with very high numbers of OODs are highlighted.

Attribution method	SST-2			IMDB			AG-News		
	pythia1	**pythia2**	**gptj**	**pythia1**	**pythia2**	**gptj**	**pythia1**	**pythia2**	**gptj**
gradnorm1	67.5	63.5	72.5	78.5	76.5	68.8	22.0	18.5	17.0
gradnorm2	66.5	61.0	71.0	75.5	80.5	67.5	22.0	19.5	17.0
gradinp	32.0	31.0	33.0	54.5	50.5	49.0	21.5	21.0	23.0
erasure	56.5	47.5	56.5	59.0	61.5	55.5	24.5	24.0	23.0
IG	18.5	17.5	23.5	33.5	31.5	41.0	16.5	14.5	21.0
KernelSHAP	22.0	17.5	22.5	30.5	34.0	31.0	18.0	13.5	18.5
Random	22.5	18.5	23.5	33.5	38.5	35.0	18.5	15.5	18.5

Table 7: The mean percentage of success in flipping Gemma-ft's label in 200 examples of evaluation split in SST-2, IMDB, and AG-News datasets (higher is better).

Attribution method	SST-2			IMDB			AG-News		
	pythia1	**pythia2**	**gptj**	**pythia1**	**pythia2**	**gptj**	**pythia1**	**pythia2**	**gptj**
gradnorm1	34.5	31.0	33.5	37.5	36.0	44.0	12.0	10.0	18.0
gradnorm2	34.0	31.5	33.5	37.0	35.0	43.5	12.0	10.0	18.5
gradinp	27.0	25.0	28.0	38.5	36.5	62.0	18.5	18.5	17.0
erasure	29.5	26.5	32.5	29.5	26.5	46.0	17.0	16.0	21.5
IG	23.5	24.5	23.0	37.0	38.0	61.0	17.5	18.5	12.0
KernelSHAP	28.0	26.0	21.5	34.0	36.0	59.5	17.5	18.0	19.0
Random	21.0	23.0	18.5	30.0	30.5	55.5	20.5	19.0	17.0

Table 8: The mean percentage of success in flipping Gemma-it's label in 200 examples of evaluation split in SST-2, IMDB, and AG-News datasets (higher is better).

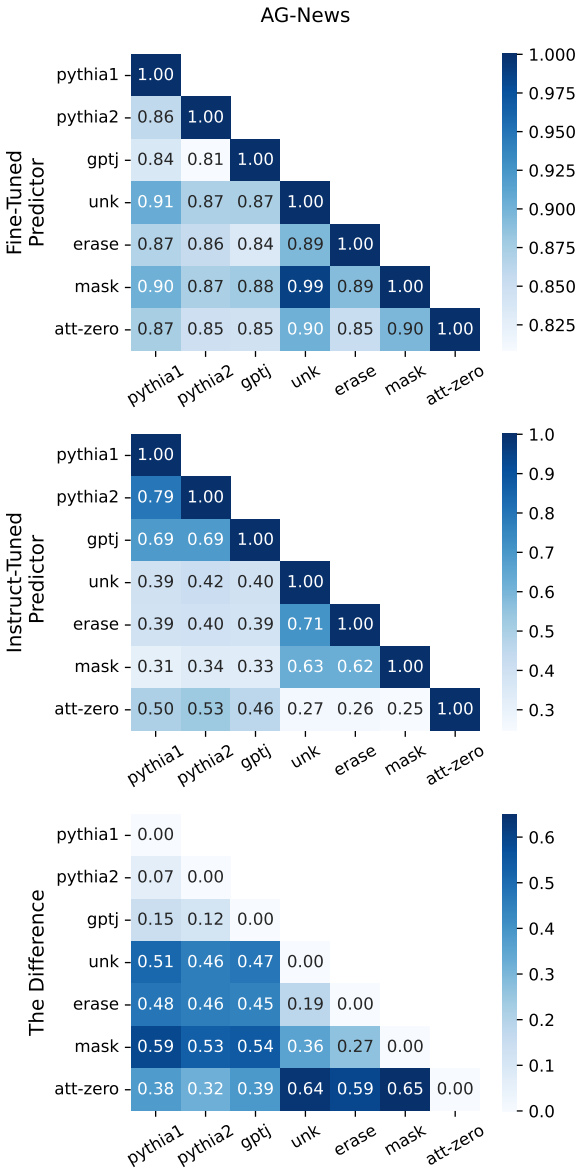

Figure 6: The difference