CAMEL: Counterfactuals As a Means for EvaLuating faithfulness of attribution methods in causal language models

Anonymous ACL submission

Abstract

Despite the widespread adoption of decoder-only autoregressive language models, explainability evaluation research has predominantly focused on encoder-only models, specifically 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. While these faithfulness evaluation techniques are suitable for MLMs, as they involve corrupted or masked inputs during pretraining, they create out-of-distribution inputs for CLMs due to the fundamental difference in 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 natural, fluent, and in-distribution counterfactuals, something that we show is important for a faithfulness evaluation method. We apply our method to several attribution methods and evaluate their faithfulness in predicting the important tokens of a few large language models.

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 a great motivation to understand their decision-making process, and so, the explanation methods are becoming increasingly important.

Attribution methods try to explain which input features are most salient in the model's predictions. A pressing issue for testing and evaluating the attribution methods is that most techniques focus on encoder-only masked language models (Modarressi et al., 2023). Almost all previous faithfulness evaluation techniques corrupt the input in one way or another, i.e., masking or erasing unimportant tokens according to the attribution technique, and then looking at the change in the prediction. These techniques work probably just fine for MLMs, pre-trained for mask and span infilling. For causal language models (CLMs) like GPT-2, pre-trained for the next token prediction, masking or erasing creates an out-of-distribution 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.

In this work, inspired by counterfactual generation—changing the model's input in a way that flips the output—we develop a technique for evaluating 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. That 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 not changing the input data distribution of the model, suitable for the attribution methods of language models. (ii) We evaluate and rank some of the more popular attribution methods using our approach. (iii) In our evaluations we find that linear and complete feature attribution methods like integrated gradients are no better than random in helping a counterfactual generator model to flip the label which is the same as when users want to infer counterfactual model behavior (Bilodeau et al., 2024).

2 Related work

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. 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). 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). Surrogate model 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). 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., 2022).

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 but 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). So 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. Hooker et al. (2019) suggest retraining the model for removed percentages of the input to achieve a model that has the same train and evaluation distribution. However, this retraining is expensive while also changing the model parameters by retraining.

Some of the previous work consider an attribution method either faithful or unfaithful but not both Han et al. (2020); Jain et al. (2020) and use the term faithful "by construction". Other works argue that faithfulness is more of a spectrum and we should evaluate the "degree of faithfulness" of an explanation method (Jacovi and Goldberg, 2020). We use the latter approach and search for a sufficiently faithful explanation method for our tasks. Ross et al. (2021) use attributions to generate counterfactuals and 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.

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.

Out-of-Distribution Detection. In order to make sure the generated counterfactuals are in-distribution we use an OOD detection method. 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. There are two common types of OOD detection methods, calibration and density estimation. density estimation methods, e.g. PPL outperform calibration methods under background shifts while the opposite is true under semantic shift. As we want to evaluate background shifts we use density-based methods. Chen et al. (2023) show that fine-tuning eliminates the pre-trained task agnostic knowledge about general linguistic properties which are useful cues for the detection of non-semantic shift. We use both fine-tuned and off-the-shelf instruct-tuned models in our evaluations to see the difference in explanation evaluation.

3 Our method

Our faithfulness evaluation protocol consists of two models. The first is the counterfactual generator model and the second is the predictor model. We want to evaluate the faithfulness of attribution methods for the predictor model. First, we give a sentence to the predictor, then use an attribution method 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 2. 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.

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 protocl to evaluate attributions consists of two phases. The first phase is creating the editor that can generate counterfactuals. We employ two approaches for creating the editor model. Our first approach is prompting a powerful off-the-shelf instruction-tuned editor to change the corrupted sentence. Our second approach is fine-tuning a smaller 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.

In the second phase of evaluating attributions, first we acquire the most important tokens according to a specific attribution method that was applied to the predictor and mask those tokens, use the second most probable prediction of the predictor between the labels as the foil label, then use one of the counterfactual generators to generate the unmasked sentence. The prompting technique used for the first approach of counterfactual generation (using off-the-shelf instruct-tuned model) during evaluation is shown in the upper part of figure 1 And the prompting technique used for the second approach of counterfactual generation (using our fine-tuned model) during evaluation is shown in the lower part of figure 1. If counterfactual generator is unsuccessful in flipping the predictor's prediction, we linearly increase the mask percentage from 10 up to 50 percent of tokens.

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

```
In the statement in backticks, replace any
<unk> with a word in a way that the resulting
statement would have a <foil label> sentiment.
output just the completed sentence.
```<masked text>```
Answer:
```

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

Figure 1: Prompting techniques used for counterfactual generation. The upper / lower box is the prompt format given to our instruct-tuned generator / fine-tuned generator.

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 and Goldberg, 2020).

4.2 Models

4.2.1 Editor Models

For editing the corrupted input we use three models. Off-the-shelf phi-3-14B-instruct(phi-it) (Abdin et al., 2024), and two models that we fine-tune in accordance with section 3 , phi-3-3.8B (phi-ft) (Abdin et al., 2024) and Pythia-2.8B (pythia) (Biderman et al., 2023) using Low Rank Adaptation (LoRA) (Hu et al., 2022). We train these models for 15 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 one-shot scenario.

4.3 Attribution Methods

Here we detail the six attribution methods that we use. 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 $\times$ 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 $\times$ Input

In gradient $\times$ input 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:

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

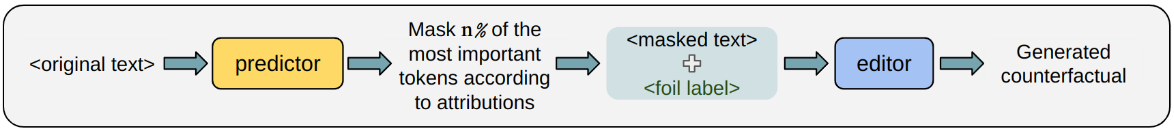

Figure 2: Our process of generating counterfactuals for evaluating attribution methods. The predictor (an LM), generates a label for the given text, and an attribution method specifics the most important tokens. We mask the top n% of them and ask an editor (another LM) to change the label of the input 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.

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:

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

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

In Tables 1 and 2, 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. In Tables 3 and 4, we show what percentage of labels each counterfactual generator is able to flip by changing the corrupted tokens for fine-tuned and instruct-tuned predictor models, respectively. 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 (Tables 1 and 3), gradient norm methods consistently perform the best for SST-2 and IMDB datasets. For AG-News, the Erasure method always performs the best or near the best. In Bilodeau et al. (2024), it is proved that for mildly rich model classes (today's language models easily surpass this richness threshold), it is impossible to conclude that the user does better than random guessing at inferring counterfactual model behaviour using linear and complete feature attribution methods without strong additional

Attribution method	SST-2 phi-it	SST-2 phi-ft	SST-2 pythia	IMDB phi-it	IMDB phi-ft	IMDB pythia	AG-News phi-it	AG-News phi-ft	AG-News pythia
gradnorm1	24.25	**34.70**	**31.60**	18.00	**29.35**	24.40	43.20	42.60	40.70
gradnorm2	**23.95**	35.70	31.75	**17.85**	29.60	**24.30**	43.30	42.75	41.35
gradinp	33.80	38.65	36.00	22.70	31.05	26.60	44.40	**42.25**	40.65
erasure	25.65	35.35	32.55	20.45	29.75	25.50	**42.90**	42.30	**40.25**
IG	35.30	41.35	41.65	32.30	35.35	30.60	46.65	44.30	42.00
KernelSHAP	32.85	41.80	40.60	30.85	35.35	30.40	46.45	43.90	42.00
Random	34.95	42.80	40.05	30.95	35.05	32.05	46.05	43.35	42.45

Table 1: 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). Off-the-shelf phi-3-14B-it (phi-it), fine-tuned pythia-2.8B (pythia), and fine-tuned phi-3-3.8B (phi-ft) models are used to fill the masks and generate counterfactuals.

Attribution method	SST-2 phi-it	SST-2 phi-ft	SST-2 pythia	IMDB phi-it	IMDB phi-ft	IMDB pythia	AG-News phi-it	AG-News phi-ft	AG-News pythia
gradnorm1	26.40	31.25	27.85	37.80	34.65	36.15	38.25	24.95	18.55
gradnorm2	26.45	30.35	28.15	38.25	34.90	34.90	38.25	24.70	18.75
gradinp	26.80	31.85	28.60	36.50	33.70	**32.95**	39.35	**24.65**	18.40
erasure	26.70	**29.05**	**23.80**	37.15	35.10	35.75	**36.95**	24.85	18.30
IG	**26.00**	30.60	26.50	37.10	33.45	34.70	39.60	25.25	**17.95**
KernelSHAP	30.00	30.90	25.50	**34.85**	**32.90**	33.80	39.40	25.70	18.10
Random	29.60	31.90	26.65	35.65	32.95	35.70	38.20	24.85	18.10

Table 2: The mean percentage needed to mask to achieve flipping Gemma-it's label or reaching 50 percent masking in one-shot scenario in 200 examples of evaluation split in SST-2, IMDB, and AG-News datasets (lower is better). Phi-3-14B-it (phi-it), fine-tuned pythia-2.8B (pythia), and fine-tuned phi-3-3.8B (phi-ft) models are used to fill the masks and generate counterfactuals.

Attribution method	SST-2 phi-it	SST-2 phi-ft	SST-2 pythia	IMDB phi-it	IMDB phi-ft	IMDB pythia	AG-News phi-it	AG-News phi-ft	AG-News pythia
gradnorm1	**87.5**	**63.5**	**72.0**	**99.0**	74.0	87.5	20.0	22.5	27.5
gradnorm2	87.0	59.0	71.5	97.5	**78.0**	**91.5**	19.0	22.5	26.0
gradinp	57.0	37.0	48.0	85.0	72.0	77.5	20.5	**24.0**	27.5
erasure	80.5	53.0	68.0	89.0	73.0	78.0	**23.0**	**24.0**	**28.0**
IG	52.5	35.0	35.0	77.5	63.0	70.0	13.0	18.5	23.5
KernelSHAP	60.0	29.5	34.0	78.0	57.0	72.5	17.0	18.5	22.0
Random	53.0	30.5	38.5	76.0	60.5	68.5	15.0	20.0	23.5

Table 3: 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). Phi-3-14B-it (phi-it), fine-tuned pythia-2.8B (pythia), and fine-tuned phi-3-3.8B (phi-ft) models are used to fill the masks and generate counterfactuals.

assumptions on the learning algorithm or data distribution. They prove this for SHAP and IG which are linear and complete, and we get a similar result that IG and KernelSHAP methods are no better than random at helping the counterfactual generator to flip the predictor model's label. Our results show that simple methods like gradnorm1, gradnorm2, and Erasure are consistently better regardless of the editor used.

For the instruct-tuned predictor (Tables 2 and 4), no attribution method is consistently better than random. This indicates that these methods are not

Attribution method	SST-2 phi-it	SST-2 phi-ft	SST-2 pythia	IMDB phi-it	IMDB phi-ft	IMDB pythia	AG-News phi-it	AG-News phi-ft	AG-News pythia
gradnorm1	69.0	56.5	65.0	38.0	47.5	44.5	41.5	65.0	83.0
gradnorm2	69.0	56.0	64.0	37.0	48.0	46.5	43.0	65.5	83.5
gradinp	69.5	52.0	62.0	49.0	54.0	**57.5**	40.5	67.0	83.5
erasure	66.0	**61.0**	**71.5**	43.0	48.5	50.5	**45.0**	**67.5**	83.5
IG	**73.0**	56.5	69.0	53.0	54.5	50.5	36.5	64.5	**84.5**
KernelSHAP	62.0	58.0	69.5	56.0	55.0	54.5	43.5	63.5	83.5
Random	62.0	52.0	69.0	**63.0**	**55.5**	52.5	43.5	66.5	83.5

Table 4: 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). Phi-3-14B-it (phi-it), fine-tuned pythia-2.8B (pythia), and fine-tuned phi-3-3.8B (phi-ft) models are used to fill the masks and generate counterfactuals.

Editor	SST-2 gemma-ft	SST-2 gemma-it	IMDB gemma-ft	IMDB gemma-it	AG-News gemma-ft	AG-News gemma-it
phi3-ft	**0.20**	10.96	4.13	4.61	1.29	5.68
phi3-it	0.40	3.11	**3.64**	**2.44**	**1.63**	**2.07**
pythia-ft	0.24	**2.85**	25.92	4.80	1.66	3.47
erase	1.17	57.28	3.10	48.02	3.44	48.0
unk	1.81	98.88	87.52	99.35	1.60	99.18

Table 5: OOD percentage when our counterfactual editor models generate samples, compared to Erase and UNK methods. The numbers are the percentages of corrupted examples that are out of the 99th percentile of the negative log likelihood of the original sentences (lower is better).

faithful for the models not fine-tuned on the task, suggesting careful application of them to the pretrained LLMs.

Why should we use counterfactuals instead of erasing the important tokens or replacing important tokens? We want to evaluate attribution methods and not the predictor model's robustness to OOD text. In order to show that we have achieved this goal, we use an OOD detection technique to measure what percentage of our generated inputs are OOD. To do so, for each dataset, we measure the negative-log-likelihood (NLL) of the 200 original text using different predictors. We evaluate masked text in three ways: (I) using an editor to fill the mask (ii) using an unimportant token (the <unk>token), and (iii) erasing the tokens. We do this test for the five amounts of corruption (10 to 50 percent) and for the seven types of attribution methods and take the average. In OOD detection a threshold is always used to label anything higher than that as OOD. We set this threshold to be the 99th percentile of the original sentences' NLL. We consider anything that has an NLL of more than this threshold as OOD.

In Table 5, we show that predictor models that are fine-tuned for classification in a specific dataset are mostly insensitive to corruption. A model that is fine-tuned for sentiment analysis becomes insensitive to unnaturalness of the text. Also, it is shown that for predictor models that are not fine-tuned for a specific dataset, corrupting the inputs makes those inputs OOD.

We design another test to understand the consistency of the editor models with each other, and the consistency of the editor models with other corruption methods. We use Spearman's rank correlation coefficient. We rank the attribution methods' mask percentage needed to flip the label for each example for all five corruption methods (our three editors, Erase, and <unk>), get the correlation of these ranks with each other, and then take the average over 200 examples. We show this for SST-2 dataset in 4. Other datasets get similar results and are shown in appendix A. The first row of figure 4, these average correlations are shown for fine-tuned models. The second row shows these correlations when the predictor models are off-the-shelf instruct-tuned predictor models. The figure shows the rank of explanations using editors has a higher correlation with unk/erase when the predictor model is fine-tuned. The third row of 4 is

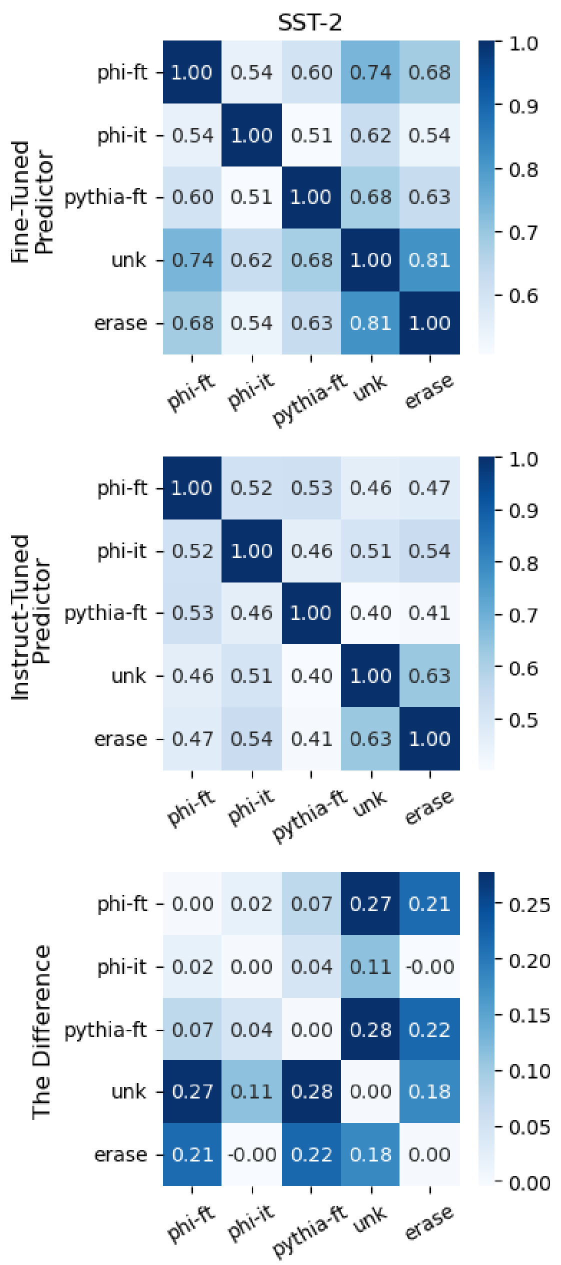

Figure 4: In the first row average correlation of attribution ranks for fine-tuned predictor is presented, and the second row presents the average correlation of attribution ranks when the predictor is an off-the-shelf model. The third row shows the difference between the first two, which shows when we use editors the difference in correlation between two different kinds of predictors is near zero but using unk/erase is not consistent in the two scenarios

the difference between the first and second rows. It indicates that the correlation difference of editors using fine-tuned and instruct-tuned predictors is near zero but the difference with other corruption methods (unk/erase) 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.

6 Conclusion

In this work we designed a faithfulness evaluation protocol based on counterfactual generation. We showed attribution methods have different efficacy in models that are fine-tuned for our specific dataset and off-the-shelf instruct-tuned models. We showed that counterfactual generators are a good option for evaluating feature attribution because they can generate mostly in-distribution text for the predictor model and the counterfactual generator is able to separate evaluating model and evaluating attribution because we are sure the examples we are evaluating the model on are mostly in-distribution. In the end we also showed that the attribution methods that are close to random in helping a user to infer counterfactual model behavior are also close to random in helping counterfactual generator to create a counterfactual.

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 it in our approach. Third, we evaluated our faithfulness method for classification datasets. Including generative tasks is more challenging in this framework. It isn't more challenging than other faithfulness evaluation methods Finally, some of the well-performing attribution methods like DecompX (Modarressi et al., 2023) are implemented for specific architecture of transformers (BERT-like models), and since our method is introduced for more recent generative models, we could not evaluate them in this paper.

References

Marah Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Qin Cai, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Yen-Chun Chen, Yi-Ling Chen, Parul Chopra, Xiyang Dai, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Victor Fragoso, Dan Iter, Mei Gao, Min Gao, Jianfeng Gao, Amit Garg, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Ce Liu, Mengchen Liu, Weishung Liu, Eric Lin, Zeqi Lin, Chong Luo, Piyush Madan, Matt Mazzola, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Xin Wang, Lijuan Wang, Chunyu Wang, Yu Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Haiping Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Jianwei Yang, Ziyi Yang, Yifan Yang, Donghan Yu, Lu Yuan, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

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.

Blair Bilodeau, Natasha Jaques, Pang Wei Koh, and Been Kim. 2024. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2):e2304406120.

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. 2022. 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.

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.

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.

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.

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.

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.

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.

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.

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.

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.

A

Figure 5 shows the difference of correlations.

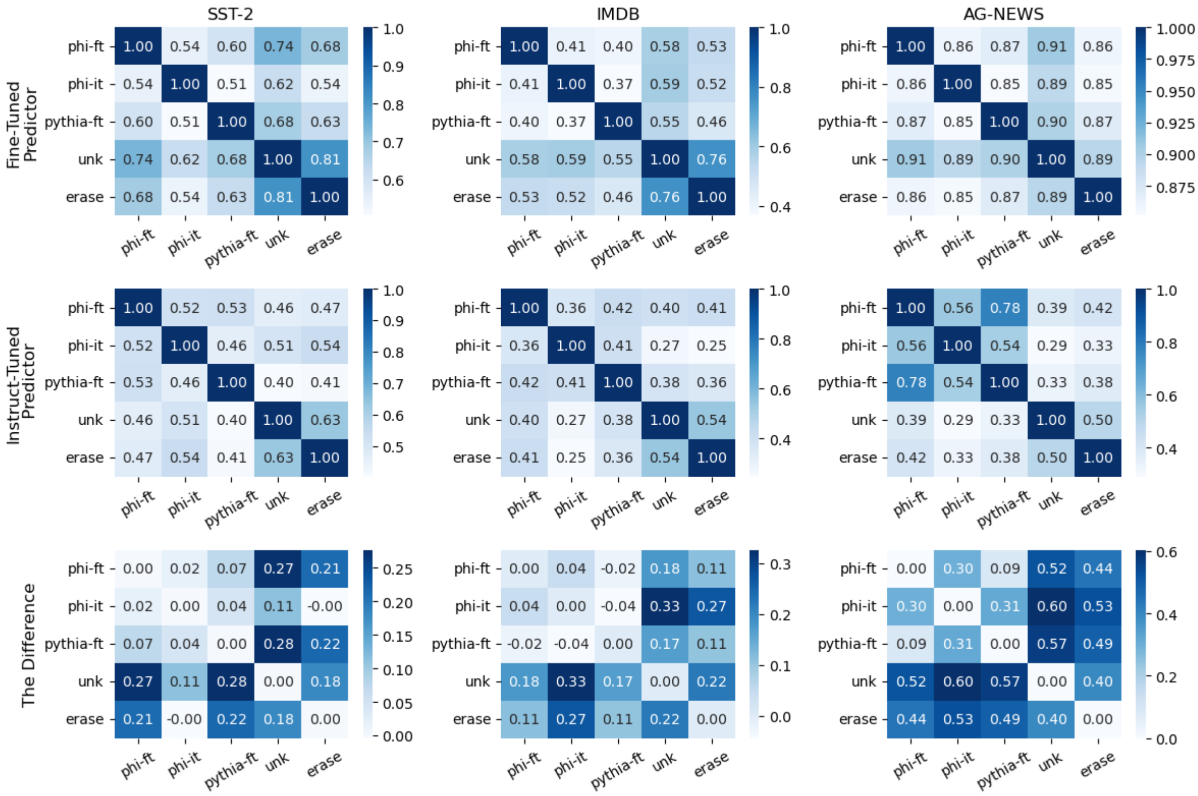

Figure 5: The difference