Master's Thesis

Improving Robustness of Pre-trained Language Models by Counterfactual Explanations

2110440 LUU, Linh Hoai

Supervisor Associate Prof. Naoya Inoue
Supervisor Prof. NGUYEN Le Minh

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

September, 2023

Abstract

Recent natural language processing (NLP) techniques have achieved high performance on various NLP benchmark datasets, primarily due to the significant improvement of deep learning. One of the approaches for improving the robustness of a model is adversarial training by adversarial examples; however, in previous adversarial training works, the adversarial examples were not guaranteed to be minimally edited and to change the model's prediction. Our hypothesis is adversarial training could make models more robust if the adversarial examples were guaranteed to be minimally edited and to change the model's prediction. In our work, we use counterfactual explanations to improve the robustness of the model. Because they are guaranteed to be minimally edited and to change the model's prediction from the original inputs. We evaluate our proposed methods in in-domain and out-of-domain settings. Experimental results show that our proposed method outperforms the pre-trained model. It indicates that using counterfactual explanation-based adversarial training to fine-tune the pre-trained model is a promising approach to improve the robustness of the pre-trained language models.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent natural language processing (NLP) techniques have achieved high performance on various NLP benchmark datasets, primarily due to the significant improvement of deep learning [21]. However, the research community has demonstrated that the NLP models are vulnerable to *adversarial attacks* [19], i.e., they are susceptible to *adversarial examples* and tend to make incorrect predictions. An adversarial example is to add some noise to the original input with the purpose of confusing a deep neural network and causing misclassification in predicting new instances. Existing pre-trained models still need to be improved for robustness. In machine learning, "robustness" is the capacity of a model to generalize successfully on new data and to handle unforeseen situations.

Adversarial training is one of the promising approaches for handling problems by generating perturbed examples of training data and additionally fine-tuning models on the perturbed examples [26], [3]. However, in previous studies on adversarial training, the generated perturbed examples were not guaranteed to be minimally edited from the original inputs and to change the model's prediction. Such perturbed examples may not be able to fool the models, leaving room for better adversarial training.

We hypothesize that adversarial training may be more effective in improving robustness when the perturbed examples are guaranteed to flip the model's prediction and to be minimally edited to change the model's prediction. In Explainable Artificial Intelligence, such perturbed examples are known as *counterfactual explanations*.

In this work, we investigate the potential of counterfactual explanations for improving the robustness of NLP models. There are several existing studies to generate counterfactual explanations [29], [34], [6], and we leverage BERT-based Adversarial Examples (BAE), one of the strong methods of adversarial attack to find a minimal edit from an input to change the model's

prediction by a masked language model-based perturbation.

To test whether counterfactual explanation helps improve the model's robustness, we setup the following pipeline. We first sample training instances that are considered less confident by a model. We then generate a set of counterfactual explanations for these unconfident examples. Finally, we fine-tune the model on the unconfident examples and these counterfactual explanations, hoping that these "edge cases" inform the model more about decision boundary.

We evaluate the counterfactual adversarial training on Natural Language Inference (NLI) and Sentiment Analysis (SA), two representative NLP tasks, in both in-domain and out-of-domain settings. Our experiments show that the model fine-tuned on counterfactual explanations outperforms the original model in both settings. Besides, we analyze the fine-tuned model's behaviors in predicting new examples and its counterfactual explanations, then compare them with the pre-trained model. Overall, the results indicate that counterfactual explanation-based adversarial training is a promising approach to improving the robustness of the pre-trained language models.

Contributions

- We introduce a new approach to adversarial training–using counterfactual explanations to improve the robustness of NLP models (§3).
- We show that the counterfactual adversarial training improves the robustness of the original model on the NLI and SA tasks, two representative NLP tasks (§4.4.1, §4.4.2).
- We provide an in-depth behavior analysis of counterfactual adversarial training (§4.4.1, §4.4.3).

Chapter 2

Background

2.1 Robustness in Natural Language Processing

Recently, there have been several approaches to improve the robustness of NLP models. Moosavi et al. [18] extend training dataset combined with their corresponding predicate-argument structures to make Transformer models understand the important parts of inputs, improving the robustness of the models. The data augmentation technique aims to increase the diversity of the training set without collecting new data.

Feng et al. [7] conduct a comprehensive survey on data augmentation, such as rule-based and example interpolation, for enhancing the robustness of models. For the rule-based approach, Wei and Zou [32] used perturbation operations, including synonym replacement, random insertion, deletion, and swap on the token level. It demonstrates that their work boosts the performance of many classification tasks and achieves strong results on small datasets.

Mixup [36] is one of the pioneers of the example interpolation approach; their proposed method aims to reduce the sensitivity to adversarial examples and unexpected behaviors such as memorization of large deep neural networks. They interpolate pairs of examples and their labels for training and regularize the neural network to prefer simple linear relationships among those training examples.

To achieve robustness, [10] proposed to train and evaluate a model adversarially using word substitutions. [11] suggest a simple but strong baseline to generate adversarial examples, which can preserve similar semantic meaning to the original input and fool well-trained model .

Alzantot et al. [1] also show that in the word substitution method, a

small token replacement can seriously change the semantics of the original input. They leverage a black-box population-based optimization method to output semantical and syntactical adversarial examples, then successfully cause misclassification for sentiment and textual models with rates of 97% and 70%, respectively.

2.2 Explainable Natural Language Processing

Recent years have seen a broader range of models and a strong improvement in the quality of state-of-the-art models, but the trade-off is that models are becoming less interpretable, and humans have no clues about how deep neural networks actually provide a decision. In fact, we do not require Artificial Intelligence (AI) models to be explainable all the time; however, we do need them in scenarios where the model's decision has a strong influence, such as in the case of a medical diagnosis.

In Explainable AI, especially Explainable NLP a model can be addressed by its scope and properties, mainly by two aspects: local or global interpretability and self-explaining or post-hoc [5].

In the first aspect, local interpretability means that the explanation is for understanding the model's behaviors for an individual sample. *LIME* [23] is one of the most impressive methods of local interpretability approach. They leverage a comprehensive model to approximate the black-box model. In this work, the authors feed the perturbed data samples into the black-box model and then observe the corresponding outputs. Because using an interpretable model, such as a linear model, requires a trade-off between accuracy and interpretability.

Global interpretability means that the entire deep network's behaviors can be comprehended by considering the parameters, weights, etc. of the model [13]. The complexity of this approach depends on the number of presented features of the data, so Honegger et al. [9] mention that it could make deep neural networks difficult to comprehend.

In the second aspect, a self-explaining approach could be considered directly interpretable; the neural networks could produce both explanation and prediction at the same time. Arya et al. [2] mention that one of the famous models of this approach is decision trees and rule-based models. While post-hoc approaches explain the model's behaviors by performing additional operations, *LIME* is one of the methods that apply post-hoc for explanation.

2.3 Counterfactual Explanations

In previous work, a counterfactual explanation is expected with the smallest number of edits in the feature that leads to the changes in the model's prediction [27], [4]. There are two main ways to generate counterfactual explanations: manually and automatically. While handcrafted methods achieve high correctness in grammar and naturalness, they could be costly. However, the automatic generation method may produce inconsistent counterfactual explanations that can not flip the model's prediction.

Several automatic methods are proposed to create counterfactual explanations and to focus on using them to explain the behavior of the black-box model.

Tomolei et al. [29] aim to transform true negative instances into positively predicted ones by using a tree-based ensemble classifier to obtain recommendations for the transformation step. Their approach proposes an algorithm called Feature Tweaking and is evaluated on Yahoo Gemini, an online advertising application.

Polyjuice framework [34] generates counterfactual explanations by leveraging GPT-2 [22]. They use the prompt format that concatenates the original input, the control code, and the masked token ([BLANK]), then fills in the [BLANK].

Elazar et al. [6] introduce an Amnesic Probing method that feeds the model with the contextualized representation of input token, then returns the output without some specific information.

Yang et al. [35] propose an approach to generate counterfactual explanations for the financial domain. Their method first has a transformer variant and is fine-tuned on a highly sensitive domain, the mergers and acquisitions prediction task. They leverage a sampled contextual decomposition technique after the prediction to calculate importance tokens, then replace them to generate counterfactual explanations.

Furthermore, "contrastive explanation" is another form of explanation but much more similar to counterfactual ones. In MiCE work [25], the purpose of "contrastive explanations" is to answer "why p not q" or which tokens make the model predict p (q). They mask and replace input tokens with others that should be fluent and similar to the original one. To choose input tokens to be masked, binary search and beam search are adopted to track the confidence of model prediction, then mask those tokens that cause the highest confidence.

In our work, we leverage counterfactual explanations to improve the robustness of the pre-trained models instead of using adversarial examples or data augmentation methods. We also require minimal edits compared with

the original input to generate counterfactual explanations; however, we mask the tokens that contribute the most to the predicted label and replace them with similar semantic ones. Besides, we sample unconfident instances for generating new examples and fine-tune the models on them to improve the robustness of models.

2.4 Transformer

In this work, we use the fine-tuned RoBERTa model, which is built on a deep learning model called Transformer. It can understand the whole context by capturing the long-term dependencies of the input text. Besides, Transformer is flexible in adapting to different tasks.

The main architecture of Transformer [30] includes an encoder and decoder (Figure 2.1).[1]

The encoder is a combination of N layers, each of which consists of two smaller sublayers. The first sublayer is a multi-head self-attention mechanism, and the second sublayer is a feed-forward network. Between each of the two sublayers are residual connections and a normalization layer.

The purpose of the decoder is to decode the input vector into the target vector, and it receives the information by vector key and value from the encoder. The decoder's architecture is almost similar to the encoder, but there is a multi-head attention in the middle that is used to learn the relationship between the selected word and other words in the input sentence.

The input of the encoder is created from a word embedding and a positional embedding. word embedding has a number of rows equal to the size of the vocabulary set. Transformer does not process input in left-to-right order like several traditional neural networks, so the model needs to remember the position of all words. Therefore, they use positional embedding, which has the same size as word embedding, to encode each word in input by a vector, then add it to word embedding.

2.4.1 Self-Attention Mechanism

Self-Attention Mechanism realizes that one word in a different sentence might have different meanings. For the below example, token *bank* in sentence (1a) means "the land alongside or sloping down to a river or lake"; while token *bank* in sentence (1b) means "a financial establishment that invests money":

(1) a. He met her when walking along the river bank.

[1] https://glassboxmedicine.com/2019/09/07/universal-transformers/

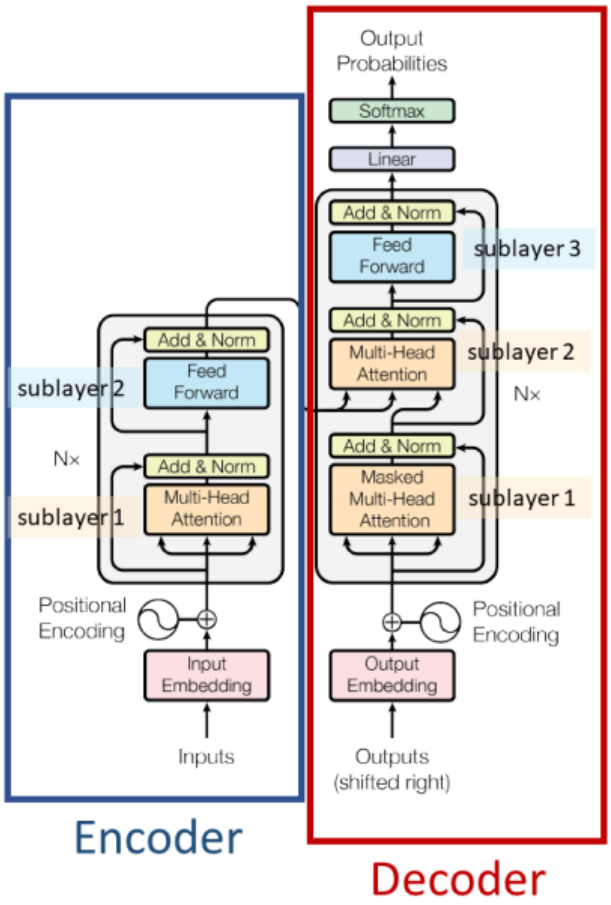

Figure 2.1: Transformer's architecture.

b. He met her when driving to the bank.

When each word is processed by the encoder, the self-attention mechanism aims to find the best representation of this word that should fit the context of each sentence or paragraph. It calculates three steps to obtain an attention vector by allowing for the consideration of context information from around words in a sentence (Figure 2.2).

In the first step, to calculate three matrices, including the query matrix W_Q, key matrix W_K, and value matrix W_V, in which W_Q, W_K and W_V are trainable weighted matrix. We multiply inputs and these matrices to get three corresponding matrices (Q, K, V). To calculate attention weights, we combine the query matrix Q and key matrix K, then use the softmax function to normalize to get the relevance between the word *bank* and others.

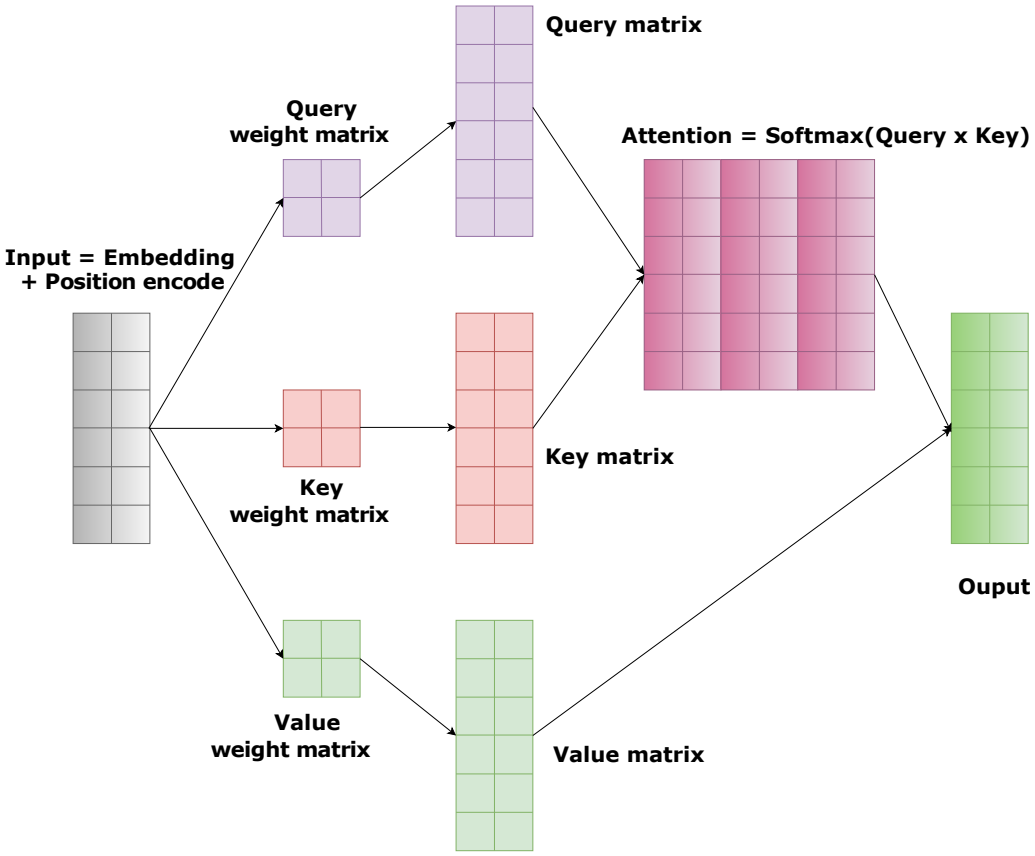

Figure 2.2: Calculating attention weights in Transformer.

Finally, we get the representation of input by multiple attention weights $\text{Attention}(Q, K)$ and a value matrix V.

2.4.2 Multi-head Attention

Multi-head attention allows the model to simultaneously pay attention to the preceding word of a word, to the next word of a word, and to the related words of a word. The model is expected to learn many types of relationships between words. For each self-attention, a pattern can be obtained, so we simply add more self-attention to help the model learn a lot of information and relationships between words thanks to different representations in different positions.

We denote that n is the number of heads, W^O is a final transformation that combines information from different heads, and W_i^Q, W_i^K, W_i^V are learnable weight matrices of Q, K, V respectively. The multi-head self-attention

of an input is calculated as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_n)W^O \tag{2.1}$$

where: $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$

Chapter 3

Proposed Method

3.1 Overview

An overview of our proposed approach is shown in Figure 3.1. The proposed method consists of three steps: (1) *sampling unconfident instances* (§3.2), (2) *generating counterfactual explanation* (§3.3), and (3) *fine-tuning pre-trained language model to improve robustness* (§3.4).

Formally, we are given (i) a dataset $\mathcal{D} = \{(x_1, y_1), ...(x_n, y_n)\}_{i=1}^{n}$, (ii) a pre-trained large language model f_θ fine-tuned for an NLP task, and (iii) a masked language model g_ϕ. In Step 1, we sample set $\mathcal{O} \subset \mathcal{D}$ of instances that are considered less confident by the black-box pre-trained model f_θ (henceforth, *unconfident examples*). These unconfident examples stand near the model's decision boundary, so the model is easy to be fooled with small edits. In Step 2, for each unconfident input $x \in \mathcal{O}_{:,1}$[1], we generate a counterfactual explanation x', yielding set $\mathcal{A}$ of counterfactual explanations. The counterfactual explanation x' of x is an example minimally edited from x that flips model's prediction. We expect that these unconfident examples and corresponding counterfactual explanations teach the model how to distinguish edge cases, which leads to the improvement of robustness of the model. In Step 3, we use both $\mathcal{O}$ and $\mathcal{A}$ to fine-tune f_θ.

3.2 Sampling Unconfident Instances

How can we obtain set $\mathcal{O}$ of unconfident examples from $\mathcal{D}$? In Active Learning, several strategies are proposed to calculate the confidence score of classification models. Suppose we have an n-class classification model parame-

[1]Following numpy notation, we denote the subscript $_{:,i}$ to denote a set of i-th element in a tuple.

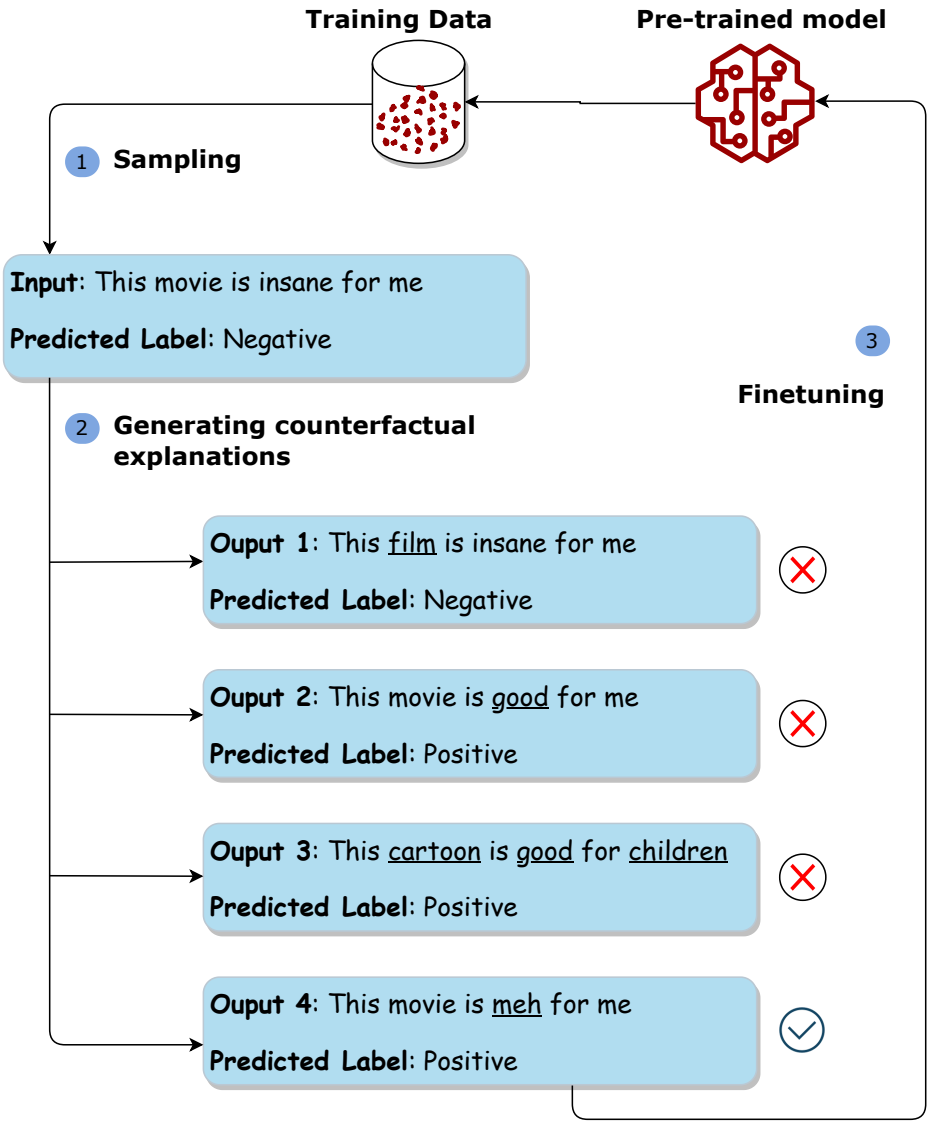

Figure 3.1: An overview of our proposed method.

terized by θ, and we are calculating the confidence score of an input x. Let $P_\theta(Y|x)$ be a probability distribution over n classes predicted by the classifier (i.e., $Y \in \{1, 2, ..., n\}$).

- Random Sampling: uses a random value as a confidence score, i.e.:

$$s_{\text{rnd}}(x) \sim \text{uniform}(0, 1) \tag{3.1}$$

  This is the most naive and commonly used method. While it preserves the original distribution of the dataset, it does not guarantee that the chosen examples will be the most unconfident one.

- Margin Confidence Sampling: calculates the difference between the top-2 prediction probabilities. Let $\hat{y}$ be a class with the highest probability, and $\hat{y}'$ be a class with the second-highest probability.

$$s_{\text{mc}}(x) = 1 - (P_\theta(Y = \hat{y}|x) - P_\theta(Y = \hat{y}'|x)) \tag{3.2}$$

- Least Confidence Sampling: returns the difference between the highest probability and 1. Let $\hat{y}$ be a class with the highest probability.

$$s_{\text{lc}}(x) = (1 - P_\theta(Y = \hat{y}|x)) \cdot \frac{n}{n-1} \tag{3.3}$$

- Entropy-based Sampling: returns the entropy of predicted probability distribution $P(Y|x)$.

$$s_{\text{ent}}(x) = -\sum_{i=1}^{n} P_\theta(Y = i|x) \cdot \log_2 P_\theta(Y = i|x) \tag{3.4}$$

We expect to sample instances that contain such a high level of "information" as to make models confused and unconfident. As in Information Theory's definition, *"the entropy of a random variable is the average level of "information", "surprise", or "uncertainty" inherent in the variable's possible outcomes."* Also inspired by how the Decision Tree method makes decisions, it uses entropy to calculate and minimize the impurity (the uncertainty) as much as possible at each leaf node. The higher the entropy is, the more informative and impurity an instance contains.

In our work, for each input $x \in \mathcal{O}$, we use Entropy-based Sampling method to calculate the confidence score of model. Suppose we have a 3-classes classification model for the NLI task. Table 3.1 shows two examples of high entropy and low entropy in the NLI task, where **Probs** column is a

Table 3.1: Example of high and low entropy predictions in NLI task. N: Neutral. C: Contradiction.

Input	Label	Probs	Entropy
Premise: Some of the unmet needs are among people who can pay, but who are deterred from seeking a lawyer because of the uncertainty about legal fees and their fear of the profession. **Hypothesis**: Some people can't afford it	N	[0.2940, 0.3706, 0.3355]	0.9959
Premise: The park is a graceful and elegant expanse with fine views of the mountains, much loved by D since first opening in 1747. **Hypothesis**: The park is ugly and you can't even see the mountains.	C	[0.9995, 0.00029, 0.00016]	0.0039

probability distribution over each class, and **Entropy** column is a normalized entropy score for the probability distribution.

For the first example, the probabilities are almost equal (0.2940, 0.3706, and 0.3355 for three classes, respectively), and the very high entropy score (0.9959) implies that the model finds unconfident to decide Neutral as the label.

In contrast, in the second example, it is easy for the model to decide that this pair of premise and the hypothesis contradict each other. The model now gives label Contradiction with a much higher score than others (0.9995), and the entropy is now tiny (0.0039), which means the model is confident about its decision.

3.3 Generating Counterfactual Explanations

Given a model f_θ and an input $x \in \mathcal{O}_{:,1}$, we generate counterfactual explanations $\mathcal{A}$ by using BERT-based Adversarial Examples (BAE) [8], which automatically generates adversarial examples using a large language model.

BAE calculates the importance score for every token of an input with respect to a target model. For token importance, they follow Jin et al.'s method [11] to inspect the difference before and after removing every token from the original one. Those highest tokens are masked, and then they leverage a large language model (BERT) to replace or insert other words. If BAE could not find new words that change the label of the newly generated

Table 3.2: Example of importance scores for each token.

	Token	Score
1	Apple	0.428319
2	matter	0.292418
3	bring	0.272522
4	will	0.267955
5	of	0.242025
6	people	0.159647
7	in	0.150222
8	future	0.120858
9	millions	0.120126
10	TV	0.095554
11	to	0.005554

example, they choose the ones that decrease the prediction probability of example the most.

In our work, we use BAE-R [8], a variant of BAE using only replacement operations. To obtain better counterfactual explanations, we make two small modifications: (i) to ensure replaced tokens have the same sentiment polarity as that of the original token, and (ii) to filter out adversarial examples that cannot change models' original predictions. We describe our method step-by-step below.

Step 1. Calculating token importance We first calculate the importance score for every token of input x and sort them in descending order into a list. To calculate the importance score, we leverage the Transformers Interpret tool[2], while what BAE did is to observe the average attention that the pre-trained language model gives to every tokens from all the layers.

For example, given an input x = *In future, Apple TV will bring matter to millions of people.*, Step 1 would produce importance scores for each token such as Table 3.2.

Step 2. Finding the best perturbation We perturb the original input x by changing important tokens one-by-one until we obtain counterfactual explanations. Each iteration consists of two processes: (i) to replace an important token $w \in x$ with a semantically similar token, and (ii) to check if the perturbed sentence x' is a counterfactual explanation.

[2]https://github.com/cdpierse/transformers-interpret

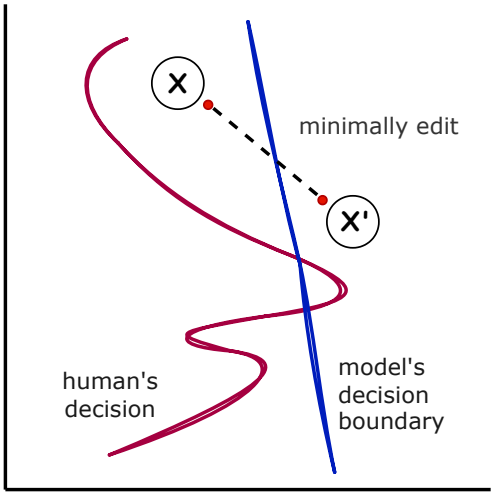

Figure 3.2: The counterfactual explanation and the original input are on different sides of the model's decision boundary, but on the same side of human's decision.

For (i), we replace the important token w with [MASK] and then use a masked language model g_ϕ (in our experiments, RoBERTa-large [14]) to predict the most-likely alternative token a for w. To ensure (i) the distance between x and the perturbed sentence is minimal (shown in Figure 3.2) and (ii) the perturbed sentence is grammatically correct, we enforce the following three constraints on a candidate alternative token a':

C1. The sentiment polarity of a' must be the same as that of w. We use SentiWordNet[3] to search and calculate sentiment score, which is a lexical resource for opinion mining with three sentiment aspects: *positivity*, *negativity*, and *neutral*.

C2. The part-of-speech (POS) of a' must be the same as that of w. We leverage nltk[4], a natural language toolkit package, to identify POS.

For (ii), we check if the perturbed sentence x' satisfies the following condition:

- $f_\theta(x') \neq f_\theta(x)$, namely x' must change the predicted label of the original input x.

If x' satisfies the condition, we terminate the process and generate x' as a counterfactual explanation. Otherwise, we iterate processes (i) and (ii)

[3]https://github.com/aesuli/SentiWordNet
[4]https://www.nltk.org/

with the next-most important token. We denote $\mathcal{A}$ as a set of generated counterfactual explanations.

Working example in sentiment analysis Suppose we are given an input *This movie is insane for me*, and a sentiment analysis model predicted NEGATIVE (i.e. $f_\theta(x) =$ NEGATIVE, shown in Figure 3.1). At Step 1, suppose *movie*, *insane*, and *me* are selected as the top-3 important tokens. We then move onto Step 2 as follows:

1. We mask *movie* (i.e. *This [MASK] is insane for me.*) and predict an alternative token with a masked language model. Suppose we obtained *This film is insane for me.*, and the model predicted NEGATIVE (i.e., $f_\theta(x') =$ NEGATIVE). Because the new modified sentence does not satisfy the above condition, we continue the process with the next important token *insane*.

2. We mask *insane* (i.e. *This film is [MASK] for me.*) and predict an alternative token with a masked language model. Suppose we obtained *This film is bad for me.*, and the model predicted POSITIVE (i.e., $f_\theta(x') =$ POSITIVE). Note that a masked language model could suggest an alternative token with positive sentiment polarity, such as *good*, but these kinds of tokens will be removed by C1. This time, because the new sentence satisfies the above condition, we terminate the process and generate *This film is bad for me.* as a counterfactual explanation.

We show three examples of generated counterfactual explanations for NLI task and SA task in Table 3.3 and Table 3.4.

We denote O represents the original input and A is the counterfactual explanation to prepare for fine-tuning the model. G column stands for the gold label of the original input. R_O, R_A column are the predicted labels by the pre-trained model for input O and counterfactual explanation A. In Table 3.3, P and H stand for premise and hypothesis, respectively for NLI task.

As we can see, the predicted labels R_A by the pre-trained language model of the couterfactual explanation examples are all changed compared to the predicted label R_O of the original input, so the pre-trained model is now fooled. Moreover, when we consider the actual labels of the counterfactual explanations E, they are still the same as the gold label G of the original inputs O.

Table 3.3: Examples of generated counterfactual explanations for NLI task. N: Neutral. E: Entailment, C: Contradiction.

Input — Counterfactual Explanation	G	R_O	R_A
O: P: Is Clinton saying he didn't commit perjury because of tthe peculiar definition of X in the Paula Jones suit, or is he saying he actually didn't have X? H: Clinton says that the X wasn't anything immoral **A**: P: Is Clinton saying he didn't commit perjury because of the peculiar definition of X in the Paula Jones suit, or is he saying he actually didn't have X? H: Clinton says that the matter wasn't anything immoral	N	N	E
O: P: What's right is wrong and what's wrong is right in some cases and it's. H: There is not telling what is right or wrong in some cases. **A**: P: What's right is wrong and what's wrong is right in some cases and it's. H: There is not telling what is right or wrong in these cases.	E	E	N
O: P: They're not on the times that i've got that i've watched because i haven't had T got TV Guide around here in ages. H: I do not have access to a TV Guide. **A**: P: I're not on the times that i've done that i've watched because i haven't had T got TV Guide around here in ages. H: I do not have access to a TV Guide	C	C	E

Table 3.4: Examples of generated counterfactual explanations for SA task. N: NEGATIVE, E: NEUTRAL, P: POSITIVE.

Input — Counterfactual Explanation	G	R_O	R_A
O: How To Dress Well play Santos Party House on October 8th with of Love and Warm Ghost! **A**: How To Dress Well plays Santos Party Hous downtown October 34th with of Love and Warm Ghost!	E	E	P
O: Photo: Blue Friday! I get to wear jeans to work and the Colts are playing tonight!! Woohoo!! Go Colts!!! **A**: Photo: Blue Dude! I need to wear jeans to work and the Colts are losing yesterday!! Woophawks!! Go Colts!	P	P	N
O: Anti-Semitic hate speech unwarranted on Iran. 80 synagogues happy with Muslims. Iraq, Iran same 1st Biblical lands. **A**: Orthodox-Semitic PC censorship unnerved on Islam. 80 synagogues happy with Muslims.Iraq, Iran same 1st Biblical lands.	N	N	E

3.4 Fine-tuning Pre-trained Language Models to Improve Robusness

Our final step is to fine-tune the pre-trained language model f_θ on both $\mathcal{O}$ and $\mathcal{A}$.

Note that $\mathcal{A}$ does not have a gold label, and it cannot be used for fine-tuning as they are. To obtain the label of $x' \in \mathcal{A}$, we use the same gold label of the original input which x' is generated from. Formally, we create a new training dataset $\mathcal{C} = \{(x', y_{\text{origin}}(x')) \mid x' \in \mathcal{A}\}$, where $y_{\text{origin}}(x)$ is the label of original input used for generating the counterfactual explanation x' in $\mathcal{O}$. We then finetune the pre-trained language model f_θ on $\mathcal{O} \cup \mathcal{C}$. We use a standard multi-class cross entropy loss for fine-tuning the model.

We expect that this can improve the robustness of the model because this teaches the model how to solve "edge" cases near the decision boundary: $\mathcal{O}$ contains a set of unconfident examples, and corresponding counterfactual explanations $\mathcal{A}$ are minimally edited examples that can fool the model.

For example, in sentiment analysis, $\mathcal{O} \cup \mathcal{C}$ may contain the following training instances:

- (*This movie is insane for me.*, NEGATIVE) $\in \mathcal{O}$
- (*This film is bad for me.*, NEGATIVE) $\in \mathcal{C}$
- (*Spielberg's movie is always exciting.*, POSITIVE) $\in \mathcal{O}$
- (*Cameron's movie is always exciting.*, POSITIVE) $\in \mathcal{C}$

where our sentiment analysis model's prediction was NEGATIVE (correct), POSITIVE (wrong), POSITIVE (correct), and NEGATIVE (wrong), respectively. This may be because the model relied on superficial cues, such as *Spielberg* $\rightarrow$ POSITIVE, changing the word *Spielberg* to something else causes a misclassification. Our counterfactual explanations are intended to fix such model's behaviors, having the model pay more attention to other important clues.

Chapter 4

Evaluation

Our main hypothesis is that counterfactual explanations could help improve the robustness of the pre-trained models. To test this, we aim to answer the following questions in two domain settings: Do counterfactual explanations help improve the accuracy of NLP models on unseen data (i) from the same domain as the training data? and (ii) from the different domain as the training data?

For the first question, we first calculate accuracy on new unseen data, and then analyze the model's behaviors to answer if fine-tuning on counterfactual explanations helps improve performance in the same domain as training data. For the second question, we also evaluate the model's performance on the data by calculating accuracy in different domains as training data.

4.1 Datasets

4.1.1 Datasets for Natural Language Inference task

To create counterfactual explanations for fine-tuning, we adopt Multi-Genre Natural Language Inference (MNLI) [33]) and use the training dataset as $\mathcal{D}$ for the NLI task. This dataset consists of 433k sentence pairs that have been crowdsourced and annotated with textual entailment understanding.

In our work, we sample from the training dataset of MNLI approximately 1200 highest entropy (low confidence) instances as dataset $\mathcal{O}$ to generate counterfactual explanations $x' \in \mathcal{A}$. The distribution of each label is shown in Table 4.1.

For in-domain evaluation, we randomly sample 550 instances from MNLI development sets ($\mathcal{O}_{dev}$) with the same distribution of each label to generate counterfactual explanation ($x_{ce} \in \mathcal{A}_{dev}$).

Table 4.1: Statistics about distribution of label for NLI task.

Label	Percentage
neutral	50.14
entailment	15.90
contradiction	33.96

Table 4.2: Statistics about distribution of label for SA task.

Label	Percentage
negative	32.24
neutral	31.01
positive	35.74

For out-of-domain datasets, we evaluate on following datasets: NLI Diagnostics, HANS, FEVER NLI, and ANLI.

- NLI Diagnostics [31] (Diagnostics) is built manually to evaluate model's performance and to analyze a variety of linguistic phenomena found in natural language.
- HANS [17] is an evaluation dataset to test unreliable syntactic heuristics that the NLI model tends to learn.
- FEVER-NLI (FEVER) is modified from the FEVER dataset [28], by utilizing a short context from Wikipedia as a premise, a hypothesis as a claim, which can be supported (entailed), refuted (contradicted), or not enough info (neutral).
- ANLI [20] is an adversarial dataset that is created from MNLI dataset, and the purpose is to make the model unconfident in predicting those samples.

4.1.2 Datasets for Sentiment Analysis task

For SA task, we experiment on the English sentiment subset from the Tweeteval dataset [24] and use the training dataset as $\mathcal{D}$ to create counterfactual explanations. This dataset consists of 59k sentences that have been crowdsourced and annotated from Twitter.

In our work, we sample from TweetEval approximately 1200 highest entropy instances (same as in NLI task) as dataset $\mathcal{O}$ to generate counterfactual explanations $x' \in \mathcal{A}$. The distribution of each label is shown in Table 4.2.

Following NLI task, we randomly sample 750 instances from Tweeteval development sets ($\mathcal{O}_{dev}$) to generate counterfactual explanation ($x_{ce} \in \mathcal{A}_{dev}$) for this task.

For out-of-domain evaluation, we evaluate on following datasets including: FinancialPhraseBank, IMDB, FiQA, StockTweet, Amazon, and Yelp.

- FinancialPhraseBank (FinP) [16] is a financial dataset including the sentiments for financial news headlines. This dataset includes approximately 5,000 sentences that are carefully annotated to ensure the business knowledge and educational background.
- IMDB [15] is a famous Internet Movie Database dataset, including 25k movie reviews labeled as positive and negative.
- FiQA[1] is a dataset from a financial domain, which is built from a microblog message, news statement, or headline.
- StockTweet[2] (Stock) is stock data collected from Twitter, including 1,300 tweets that are manually screened and reviewed.
- Amazon [12] is a product English reviews dataset containing the review text and the star rating from one to five by users. The subset used in this task is selected from multilingual text classification.
- Yelp [37] is a restaurant review dataset containing ratings from zero to four, which stand for the satisfaction level of customers.

4.2 Evaluation Metrics

Our metric is to calculate and compare the accuracy of each model in predicting an input's label compared with its gold label. Given x_{dev} as an input, we sequentially use the pre-trained model and our fine-tuned model to predict x_{dev}'s label, then compare it with its gold label. Similarly, we calculate the accuracy of each model when x_{ce} is given as an input.

Our metric is also to calculate and compare the accuracy of a model in predicting an input's and its counterfactual explanation's label. we evaluate the performance of the pre-trained model and our fine-tuned model in predicting a pair of inputs x_{dev} and their counterfactual explanations x_{ce}. Given x_{dev} as an input and its counterfactual explanation x_{ce}, we use the pre-trained

[1]https://sites.google.com/view/fiqa/

[2]https://ieee-dataport.org/open-access/stock-market-tweets-data

model to predict the two labels, then compare them together. Similarly, we our fine-tuned model to predict and compare these two predicted labels.

For each dataset in out-of-domain evaluation, we use the pre-trained model and our fine-tuned model to predict labels. We calculate accuracy that divides the number of true predicted labels by the total predicted labels, then compare the two models together.

4.3 Model Setting

Our experiments are conducted in the Pytorch framework[3] and in a large computing server with the following environment:

- CPU: Intel Xeon GOLD 5320 2.2GHz 26Core x2 Sockets: 52Cores.
- Memory: DDR4/3200 SDRAM x16 : 512GB.
- GPU: NVIDIA A100 x 2.
- Number of nodes: 10 Nodes.

4.3.1 NLI task

For the pre-trained model f_θ, we employed the RoBERTa-large model fine-tuned on the MNLI corpus.[4] Before fine-tuning models, we do a pre-processing step by setting their maximum length to 512, then pad and truncate if they exceed the limitation setup. Moreover, in the NLI task, the input is a pair of a premise and a hypothesis, so we insert tokens [CLS] at the first and [SEP] at the end to separate these two sentences.

For example, given a Premise (2a) and Hypothesis (2b), we get an Input (2c) after pre-processing:

(2) a. Premise: She is walking in the garden with her cats.

b. Hypothesis: Her cats are in the garden.

c. Input: [CLS] She is walking in the garden with her cats. [SEP] Her cats are in the garden. [SEP]

[3]https://pytorch.org/

[4]https://huggingface.co/roberta-large-mnli

Table 4.3: Hyper-parameter settings for NLI task.

Hyper-parameter	**Setting**
Batch Size	8
Warm up step	50
Optimizer	AdamW
Training Epoch	5
Learning Rate	1e-3 1e-5 1e-8

Table 4.4: Hyper-parameter settings for SA task.

Hyper-parameter	**Setting**
Batch Size	8
Warm up step	50
Optimizer	AdamW
Training Epoch	5
Learning Rate	1e-3 1e-5 1e-7

The configuration settings of the model are shown in Table 4.3. For learning rate, we setup three candidates 1e-3, 1e-5, and 1e-8 and choose the one that achieves the best result overall. We also evaluate at the end of each epoch during training and save two checkpoints, including the last one and the best one, if they are different.

4.3.2 SA task

For the pre-trained model f_θ, we use RoBERTa-base model trained on the Twitter dataset.[5] In SA task, we also do a pre-processing step for input sentences as the NLI task, but we do not insert more tokens before supplying them to fine-tune because they are single sentences.

The configuration of the model is shown in Table 4.4. The hyper-parameters is almost the same as the setting of NLI task, but we setup different learning rate 1e-3, 1e-5, and 1e-7 to choose the best one.

[5]https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment

Table 4.5: In-domain evaluation.

	$\mathcal{O}_{dev}$
roberta-mnli	90.01
fine-tuned roberta-mnli	**91.44**
roberta-twitter	74.2
fine-tuned roberta-twitter	**77.15**

Table 4.6: The percentage of the predicted label of original input and generated counterfactual explanations by the pre-trained language model and fine-tuned model that are the same as the gold label.

	$\mathcal{O}_{dev}$	$\mathcal{A}_{dev}$
roberta-mnli	90.35	9.65
fine-tuned roberta-mnli	90.16	**15.12**
roberta-twitter	74.36	22.96
fine-tuned roberta-twitter	75.57	**41.92**

4.4 Results

The results are shown in Tables 4.5–4.15. For NLI task, *roberta-mnli* stands for the pre-trained model f_θ, and *roberta-twitter* stands for the pre-trained model f_θ in SA task.

4.4.1 In-domain Evaluation

Table 4.5 shows the result of in-domain evaluation. After fine-tuning, our proposed method (*fine-tuned roberta-mnli*) gets better performance than the pre-trained model (*roberta-mnli*) by about 1.44% for the NLI task. For the SA task, we achieve approximately 3% higher when evaluating on the development set $\mathcal{O}_{dev}$.

Fine-tuned models precisely target the inconsistent gold labels' problem without hurting performance Table 4.6 shows the percentage of the predicted label of the original input x_{dev} ($\in \mathcal{O}_{dev}$) by the pre-trained models and our fine-tuned models that are the same as the gold label. Similarly, we show the result for the generated counterfactual explanation x_{ce} ($\in \mathcal{A}_{dev}$), these x_{ce} are generated from x_{dev} as mentioned in §4.2.

For the NLI task, this percentage is an insignificant difference between *roberta-mnli* and *fine-tuned roberta-mnli* (90.35% and 90.16%) when we predict the original inputs from $\mathcal{O}_{dev}$ and compare with their gold labels. How-

Table 4.7: The sensitivity of the pre-trained language model and our fine-tuned model to counterfactual explanations.

	True	**False**
roberta-mnli	2.01	97.99
fine-tuned roberta-mnli	**10.75**	89.75
roberta-twitter	3.47	96.53
fine-tuned roberta-twitter	**40.05**	59.95

ever, for generated counterfactual explanations $\mathcal{A}_{dev}$, our *fine-tuned roberta-mnli* model outperforms *roberta-mnli* by more than 5%.

Correspondingly, for the SA task, we also compare the pre-trained *roberta-twitter* model and our *fine-tuned roberta-twitter* model and observe that our model is even over 1% higher for $\mathcal{O}_{dev}$. Besides, when comparing the predicted labels of generated counterfactuals $\mathcal{A}_{dev}$ with their gold labels, our *fine-tuned roberta-twitter* model gains an impressive improvement, approximately 20% higher than the original *roberta-twitter* model.

The above result implies that our fine-tuned model precisely targets the inconsistent gold labels' problem. It does not hurt in-domain performance and sometimes it slightly increases the accuracy.

Fine-tuned model is more difficult to be fooled Table 4.7 shows the percentage of the predicted label by the pre-trained model and our fine-tuned model of the original inputs are the same as their generated counterfactual explanations.

Generally, our fine-tuned models achieve much better results than the original pre-trained models; while *fine-tuned roberta-mnli* model improves true prediction by more than 8%, the *fine-tuned roberta-twitter* model decreases false prediction by around 36.5%.

The above results reveal that after fine-tuning, our fine-tuned model is not easy to be fooled by new counterfactual explanations and to be more robust to them.

For in-domain evaluation, when fine-tuning models on counterfactual explanations, it helps to increase accuracy on the development dataset. For an in-depth analysis of the model's behavior, we observe that our fine-tuned models are more consistent in producing gold labels. Besides, they are more robust to newly generated counterfactual examples due to a much greater improvement in accuracy.

4.4.2 Out-of-domain Evaluation

Table 4.8: Out-of-domain evaluation of the original roberta-mnli model and our fine-tuned model.

Model	**ANLI**	**Diagnostics**	**FEVER**	**HANS**
	3200	1104	20k	30k
roberta-mnli	31.8	66.39	70.7	73.13
fine-tuned roberta-mnli	32.37	66.49	70.87	73.75

Table 4.9: Out-of-domain evaluation of the original roberta-twitter model and our fine-tuned model.

Model	**IMDB**	**FiQA**	**Stock**	**Yelp**	**FinP**	**Amazon**
	25k	675	1300	50k	4846	5k
roberta-twitter	77.1	76.59	55.15	68.8	67.51	69.38
fine-tuned r-tw	78.24	78.37	**57.84**	69.31	**69.17**	69.98

Tables 4.8 and 4.9 show the results of out-of-domain evaluation for NLI and SA tasks, respectively. For the NLI task, our fine-tuned model gains a minor improvement (roughly 1%) for all datasets compared to the original *roberta-mnli* model. Our fine-tuned roberta-mnli achieves the best on the HANS dataset and ANLI dataset. Besides, it insignificantly improve on the NLI Diagnostics and Fever NLI dataset. For the SA task, our fine-tuned roberta-twitter model (*fine-tuned r-tw*) gets better performance (about 2%) for most out-of-domain datasets compared to the original *roberta-twitter* model.

Our *fine-tuned roberta-twitter* model performs the best on financial datasets including Stock Twitter, FiQA, and Finacial PhraseBank datasets; however, for review datasets including Yelp, Amazon, and IMDB review datasets, it achieves smaller results (0.6-1.14%).

The above results indicate that after fine-tuning on counterfactual explanations, our fine-tuned models outperform the pre-trained models not only in the same domain but also in the different domain as training data.

4.4.3 Case Study

In this section, we give examples to compare the pre-trained models and our fine-tuned models' behaviors in predicting labels. In each table, we show the cases that the pre-trained model and our fine-tuned model predict both

correctly and incorrectly.

We denote O as the original input, A is the counterfactual explanation; G is the gold label, R_O is the predicted label of the original input x_{dev} by the pre-trained model and R_A is the predicted label of the counterfactual explanation x_{ce} by the pre-trained model; similarly, F_O and F_A is the predicted labels by our fine-tuned model.

Examples of analyzing model's behaviors in predicting the label of an input and compared with its gold label For SA task, we show examples in Table 4.10 and Table 4.11. For NLI task, we show examples in Table 4.13 and 4.12.

We show examples to cover the cases that observe the changes in the model's behavior in predicting in-domain input before and after fine-tuning. We compare the predicted labels of each models (R_O, F_O for in-domain input and R_A, F_A for counterfactual) with gold label (G).

In Table 4.10, we show examples 1 and 2 that: before fine-tuning, the pre-trained model produces different label NEUTRAL as the gold label (NEGATIVE in example 1 and POSITIVE in example 2). However, after fine-tuning, our model predicts correctly as gold label. Example 4 and 5 show that our fine-tuned model failed in prediction. While example 3 shows that our fine-tuned model still can not produce a correct label as gold label, it remains the same prediction as before fine-tuning.

Similar, in Table 4.11, we show five examples to cover five cases that observe the changes in the model's behavior in predicting counterfactual explanations before and after fine-tuning. We compare the predicted labels of each models (R_A, F_A) with gold label (G).

Table 4.10: Examples of the changes in the model's behavior in predicting in-domain input before and after fine-tuning for SA task.

	Input	G	R_O	F_O
1	I saw Paper Towns on Friday, it was good I guess but there was so much changes compares to the book, it bothered me	N	E	N
2	@user @user @user No he not Kendrick is the best rapper in mainstream right now Black Friday freestyles ex.	P	E	P
3	that's cool kind of like Pink Floyd or something uh yeah basketball's cool but football kind of after a while. Yeah, I love basketball and football	E	N	N
4	While everyone is out in the freezing cold Trick or Treating\u02c I sit here watching White Collar. Yes\u02c this is how I spend my October 31st.	E	E	P
5	@user @user @user Quote of the night- If I were Amarosa I would go to Brooklyn bridge and be drowning myself funny	P	E	N

Table 4.11: Examples of the changes in the model's behavior in predicting counterfactual explanation before and after fine-tuning for SA task.

	Counterfactual Explanation	G	R_A	F_A
1	When Millennials become bandwagon fans of the Packers because of Harry. Do y-all even know who Aaron Rodgers is? Or what a 1st down is?	N	E	N
2	@user Yeah I think so. We saw Suarez score up near us and we played pretty evenly 2nd half so it wasn \u2019t so bad. Probably should\u2019ve had ET	P	E	P
3	I've said all along and still staunchly maintain that the Paris attacks: Charlie Hebdo and the 13 November 2015 one were both Gladio propaganda	N	N	N
4	Real Madrid reportedly rule out signing De Gea in January; making me believe they're getting hot feet about this deal altogether.	E	E	N
5	My Sunday nights hadn\u2019T been the same since @user has been gone from Breakout Kings. I wonder what he\u2019s doing next? Can\u2t wait 2 c!	P	E	N

Table 4.12: Examples of the changes in model's behavior in predicting in-domain input before and after fine-tuning for NLI task.

	Input	G	R_O	R_O
1	P: get something from from the Guess Who or. H: Get something from someone or the Guess Who if you really want.	N	E	N
2	P: so well i think we've taken up at least five minutes. H:You've taken up the last 5 minutes.	N	E	N
3	P: Most produce is locally grown, with some from the restaurant's own organic garden. H: All of the produce comes from Mexico.	C	C	C
4	P: Yes, undoubtedly the hand of Mr. Brown! Mr. Carter paused. H: No identity could be assigned to the severed appendage.	C	C	N

Table 4.13: Examples of the changes in model's behavior in predicting counterfactual explanation before and after fine-tuning for NLI task.

	Counterfactual Explanation	G	R_A	F_A
1	P: You will need-all of you will need-to be highly visible personally and professionally. H: Everyone needs to maintain an open look to the public in order to attain success.	N	E	N
2	P: Participation in the rulemaking process requires (1) the public to be aware of opportunities to participate and (2) systems that will allow agencies to receive comments in an efficient and effective manner. H: The public need only be made aware of any opportunities for rulemaking processes.	C	E	C
3	P: that's cool kind of like Pink Floyd or something uh yeah basketball's cool but football kind of after a decade H: Yeah, I love basketball and football.	N	N	N
4	P: today it runs lined with shipments, factories, and industrial development, and its waters are badly cleaned. H: Its products are pure nor safe to drink	E	N	E
5	P: British action wouldn't have failed. H: British action would have made a big difference.	C	E	N

Examples of analyzing model's behaviors in predicting the label of counterfactual explanations For the NLI task, we show examples in Table 4.14 and Table 4.15. For SA task, we show examples in Table 4.16 and 4.17.

We show examples to cover the cases that observe the changes in the model's behavior in predicting in-domain input and its counterfactual explanation before and after fine-tuning. We compare the predicted labels of each models R_O, R_A and F_O, F_A.

In Table 4.14 and Table 4.15, we show examples 1 and 2 that: before fine-tuning, the pre-trained model produces different label NEUTRAL and (ENTAILMENT for an in-domain input and its counterfactual explanation. However, after fine-tuning, the model predicts the same label. Example 3 shows that our fine-tuned model failed in prediction. Before fine-tuning, the in-domain input and its counterfactual explanation have label CONTRADICTION, but after fine-tuning, they become CONTRADICTION and NEUTRAL.

The number of such failure cases is completely small, the NLI task has only two cases and the SA task has four cases.

Table 4.14: Examples of the predicted label of the original input and its counterfactual explanation by the pre-trained model for NLI task.

	Input — Counterfactual Explanation	R_O	R_A
1	**O**: P: We did it with the aid of consultants and other equal justice stakeholders. H: The help from consultants and other stakeholders was useful in doing it. **A**: P: We did it with the intervention of consultants and other equal justice stakeholders. H: The help from consultants and other stakeholders was useful in doing it.	N	E
2	**O**: P: Participation in the rulemaking process requires (1) the public to be aware of opportunities to participate and (2) systems that will allow agencies to receive comments in an efficient and effective manner. H: The public need not be made aware of any opportunities for rulemaking processes. **A**: P: Participation in the rulemaking process requires (1) the public to be aware of opportunities to participate and (2) systems that will allow agencies to receive comments in an efficient and effective manner. H: The public need only be made aware of any opportunities for rulemaking processes.	C	E
3	**O**: P: Today it is lined with shipyards, factories, and industrial development, and its waters are badly polluted. H: Its waters are pure and safe to drink **A**: P: today it runs lined with shipments, factories, and industrial development, and its waters are badly cleaned. H: Its products are pure nor safe to drink	C	C

Table 4.15: Examples of the predicted label of the original input and its counterfactual explanation by our fine-tuned model for NLI task.

	Input — Counterfactual Explanation	F_O	F_A
1	**O**: P: We did it with the aid of consultants and other equal justice stakeholders. H: The help from consultants and other stakeholders was useful in doing it. **A**: P: We did it with the intervention of consultants and other equal justice stakeholders. H: The help from consultants and other stakeholders was useful in doing it.	N	N
2	**O**: P: Participation in the rulemaking process requires (1) the public to be aware of opportunities to participate and (2) systems that will allow agencies to receive comments in an efficient and effective manner. H: The public need not be made aware of any opportunities for rulemaking processes. **A**: P: Participation in the rulemaking process requires (1) the public to be aware of opportunities to participate and (2) systems that will allow agencies to receive comments in an efficient and effective manner. H: The public need only be made aware of any opportunities for rulemaking processes.	C	C
3	**O**: P: Today it is lined with shipyards, factories, and industrial development, and its waters are badly polluted. H: Its waters are pure and safe to drink **A**: P: today it runs lined with shipments, factories, and industrial development, and its waters are badly cleaned. H: Its products are pure nor safe to drink	C	N

Table 4.16: Examples of the changes in model's behavior in predicting in-domain inputs and its counterfactual explanations before and after fine-tuning for SA task.

	Input — Counterfactual Explanation	R_O	R_A
1	**O**: Why wait for the videos when you can come see duels official live tomorrow? I may have vodka on my person too! Ha **A**: Why cry for the videos when you can come see reels official live tomorrow? I may have vodka on my person too! Ha	E	P
2	**O**: When girls become bandwagon fans of the Packers because of Harry. Do y'all even know who Aaron Rodgers is? Or what a 1st down is? **A**: When Millennials become bandwagon fans of the Packers because of Harry. Do y-all even know who Aaron Rodgers is? Or what a 1st down is?	N	E
3	**O**: @user Yeah I think so. We saw Suarez score up near us and we played pretty well 2nd half so it wasn$\breve{2}$019t so bad. Probably should$\breve{2}$019ve had ET **A**: @user Yeah I think so. We saw Suarez score up near us and we played pretty evenly 2nd half so it wasn$\breve{2}$019t so bad. Probably should$\breve{2}$019ve had ET	P	E
4	**O**: @user follow @user and meet her on NOVEMBER 26 $\breve{0}$02c she$\breve{2}$019ll be at the bell center by 4:00 pm. PLEASE FREDO. **A**: @ user follow @girl and meet her in NOOVEMBER 2019ø002c she$\breve{2}$019ll be around the bell center by 4:00 pm. PLEASE FROXO.	E	E
5	**O**: men tomorrow you will have one of your hardest patrols...CIF turn in lets hope i have everything **A**: men tomorrow you will have one of your hardest patrols...CIF turn in lets hope i save everything	E	N

Table 4.17: Examples of the changes in model's behavior in predicting in-domain inputs and its counterfactual explanations before and after fine-tuning for SA task.

	Input — Counterfactual Explanation	F_O	F_A
1	**O**: Why wait for the videos when you can come see duels official live tomorrow? I may have vodka on my person too! Ha **A**: Why cry for the videos when you can come see reels official live tomorrow? I may have vodka on my person too! Ha	P	P
2	**O**: When girls become bandwagon fans of the Packers because of Harry. Do y'all even know who Aaron Rodgers is? Or what a 1st down is? **A**: When Millennials become bandwagon fans of the Packers because of Harry. Do y-all even know who Aaron Rodgers is? Or what a 1st down is?	N	N
3	**O**: @user Yeah I think so. We saw Suarez score up near us and we played pretty well 2nd half so it wasn2̆ 019t so bad. Probably should2̆019ve had ET **A**: @user Yeah I think so. We saw Suarez score up near us and we played pretty evenly 2nd half so it wasn2̆019t so bad. Probably should2̆019ve had ET	P	P
4	**O**: @user follow @user and meet her on NOVEMBER 26 0̆02c she2̆019ll be at the bell center by 4:00 pm. PLEASE FREDO. **A**: @ user follow @girl and meet her in NOOVEMBER 2019ø002c she2̆019ll be around the bell center by 4:00 pm. PLEASE FROXO.	E	P
5	**O**: men tomorrow you will have one of your hardest patrols...CIF turn in lets hope i have everything **A**: men tomorrow you will have one of your hardest patrols...CIF turn in lets hope i save everything	P	N

Chapter 5

Conclusion and Future Work

5.1 Conclusion

To improve the robustness of models, adversarial training is one of the promising approaches. However, in previous studies of adversarial training, the generated adversarial examples were not guaranteed to be minimally edited and to change the model's prediction from the original inputs. Our hypothes is that adversarial training might be more effective in enhancing robustness if given limitations are addressed. In this work, we leverage counterfactual explanations to improve the model's robustness. Experimental results demonstrate that our proposed method outperforms the pre-trained model in both in-domain and out-of domain settings. We also explore the fine-tuned model's behaviors in its prediction compared with the pre-trained model. It indicates that counterfactual explanation-based adversarial training is a promising approach to improve the robustness of the pre-trained language models.

5.2 Future Work

In the future, we plan to combine our work with active learning to have humans check the quality of newly generated counterfactual explanations and verify the gold label of the counterfactual explanations. Besides, we aim to leverage a large language model such as GPT to generate counterfactual explanations and then compare them with our methods. Additionally, in this work, we only use one method to generate counterfactual explanations, so we want to improve counterfactual explanation generation methods. Finally, we will analyze the model's attention for an input before and after fine-tuning on counterfactual explanations.

Bibliography

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

[2] Vijay Arya, Rachel KE Bellamy, Pin-Yu Chen, Amit Dhurandhar, Michael Hind, Samuel C Hoffman, Stephanie Houde, Q Vera Liao, Ronny Luss, Aleksandra Mojsilović, et al. 2019. One explanation does not fit all: A toolkit and taxonomy of ai explainability techniques. *INFORMS 2021.*

[3] Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. 2021. Recent advances in adversarial training for adversarial robustness. *International Joint Conferences on Artificial Intelligence Organization.*

[4] Brian Barr, Matthew R Harrington, Samuel Sharpe, and C Bayan Bruss. 2021. Counterfactual explanations via latent space projection and interpolation. *CoRR*, abs/2112.00890.

[5] Marina Danilevsky, Kun Qian, Ranit Aharonov, Yannis Katsis, Ban Kawas, and Prithviraj Sen. 2020. A survey of the state of explainable ai for natural language processing. *In Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, pages 447–459, Suzhou, China. Association for Computational Linguistics.*

[6] Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175.

[7] Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 968–988, Online. Association for Computational Linguistics.*

[8] Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

[9] Milo Honegger. 2018. Shedding light on black box machine learning algorithms: Development of an axiomatic framework to assess the quality of methods that explain individual predictions. *ArXiv.*

[10] Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. *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 4129–4142, Hong Kong, China. Association for Computational Linguistics.*

[11] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In *Proceedings of the AAAI conference on artificial intelligence*, 05, pages 8018–8025.

[12] Phillip Keung, Yichao Lu, György Szarvas, and Noah A Smith. 2020. The multilingual amazon reviews corpus. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing.*

[13] Zachary C Lipton. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

[14] 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. *CoRR.*

[15] Andrew 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.

[16] Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

[17] R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.*

[18] Nafise Sadat Moosavi, Marcel de Boer, Prasetya Ajie Utama, and Iryna Gurevych. 2020. Improving robustness by augmenting training sentences with predicate-argument structures. *CoRR.*

[19] John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations.*

[20] Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4885–4901, Online. Association for Computational Linguistics.*

[21] Marwan Omar, Soohyeon Choi, DaeHun Nyang, and David Mohaisen. 2022. Robust natural language processing: Recent advances, challenges, and future directions. *IEEE Access.*

[22] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

[23] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

[24] Sara Rosenthal, Noura Farra, and Preslav Nakov. 2019. Semeval-2017 task 4: Sentiment analysis in twitter. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pages*

502–518, Vancouver, Canada. Association for Computational Linguistics.

[25] Alexis Ross, Ana Marasović, and Matthew E Peters. 2020. 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.*

[26] Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. 2019. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32.

[27] Dylan Slack, Anna Hilgard, Himabindu Lakkaraju, and Sameer Singh. 2021. Counterfactual explanations can be manipulated. *Advances in neural information processing systems*, 34:62–75.

[28] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER).*

[29] Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. 2017. Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 465–474.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

[31] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.*

[32] Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

[33] Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.*

[34] Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S 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.*

[35] Linyi Yang, Eoin M Kenny, Tin Lok James Ng, Yi Yang, Barry Smyth, and Ruihai Dong. 2020. Generating plausible counterfactual explanations for deep transformers in financial text classification. *Proceedings of the 28th International Conference on Computational Linguistics.*

[36] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. 2017. mixup: Beyond empirical risk minimization. *International Conference on Learning Representations.*

[37] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.