CAT: Enhancing Language Model Robustness via Counterfactual Adversarial Training

Hoai Linh Luu **Naoya Inoue**
Japan Advanced Institute of Science and Technology
{s2110440,naoya-i}@jaist.ac.jp

Abstract

One of the approaches for improving the robustness of NLP models is adversarial training by adversarial examples. However, in previous work on adversarial training, 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. We propose Counterfactual Adversarial Training (CAT), which uses counterfactual explanations to improve the robustness of the model. Our experiments on Natural Language Inference and Sentiment Analysis show that CAT significantly enhances out-of-the-box pre-trained NLP models on 11 datasets, indicating that CAT is a promising approach to improve the robustness of the pre-trained language models.

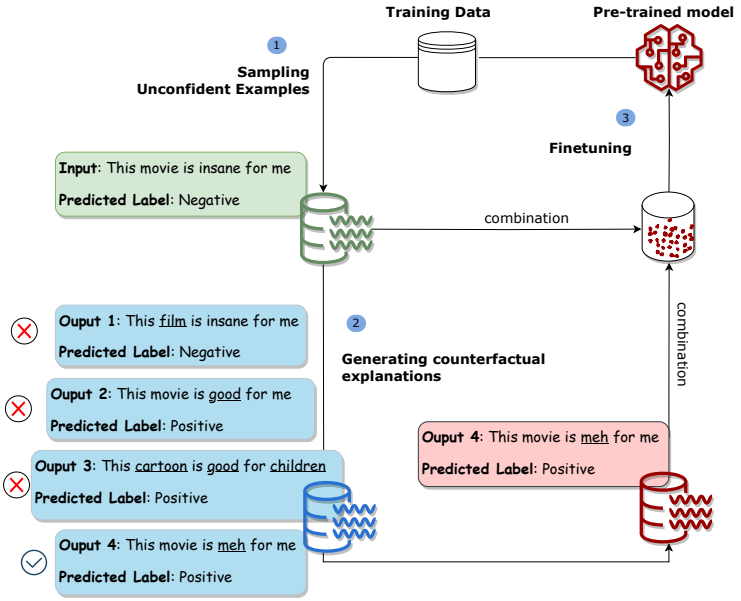

Figure 1: An overview of our proposed method.

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 (Omar et al., 2022). However, the research community has demonstrated that the NLP models are vulnerable to *adversarial attacks* (Moosavi et al., 2020), i.e., they are susceptible to *adversarial examples* and making 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, 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 improving the robustness of NLP models by generating perturbed examples of training data and additionally fine-tuning models on the perturbed examples (Shafahi et al., 2019; Bai et al., 2021). 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 AI, such perturbed examples are known as *counterfactual explanations*.

In this work, we investigate the potential of *Counterfactual Adversarial Training (CAT)*, which uses counterfactual explanations for improving the robustness of NLP models. There are several existing studies to generate counterfactual explana-

tions (Wu et al., 2021; Elazar et al., 2021), 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 CAT 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 their 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.

Our contributions are summarized as follows:

- We introduce Counterfactual Adversarial Training (CAT), a new approach to adversarial training–using counterfactual explanations to improve the robustness of NLP models (§3).
- We show that CAT improves the robustness of the original model on the NLI and SA tasks, two representative NLP tasks (§4.3.1, §4.3.2).
- We provide an in-depth behavior analysis of CAT (§4.3.1).

2 Related work

Robustness Recently, there have been several approaches to improve the models' robustness. Moosavi et al. (2020) combine the training set with their corresponding predicate-argument structures to make the transformer model understand the important parts of inputs, improving the robustness of the models. Data augmentation techniques aim to increase the diversity of the training set without collecting new data. Feng et al. (2021) conduct a comprehensive survey on data augmentation, including rule-based, example interpolation, and model-based techniques for enhancing the models' robustness.

Generating adversarial examples is considered an effective means to achieve robustness. A method is proposed to train and evaluate the model adversarially using word substitutions (Jia et al., 2019). Jin et al. (2020) suggest a simple but strong baseline to generate adversarial examples, which can preserve similar meaning to the original input and lead the model to misclassify.

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 (Slack et al., 2021; Barr et al., 2021). There are two main ways to generate counterfactual explanations: manually and automatically. While handcrafted counterfactual explanations 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 methods are proposed to create counterfactual explanations and focus on explaining the behavior of the black-box model. Elazar et al. (2021) introduce an Amnesic Probing method that feeds the model with the contextualized representation of the inputs, then returns the output without some specific information.

Polyjuice framework (Wu et al., 2021) generates counterfactual explanations by leveraging the ability of the large language model GPT-2 (Radford et al., 2019). They use the prompt format that concatenates the original input, the control code, and the masked token ([BLANK]), then fills in the [BLANK].

Moreover, *contrastive explanation* is another form of explanation but much more similar to counterfactual ones. In MiCE work (Ross et al., 2020), they are to answer *why p not q?* or which tokens make the model predict p (q). To generate counterfactuals, their inputs are masked and flipped labels. To choose a token 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

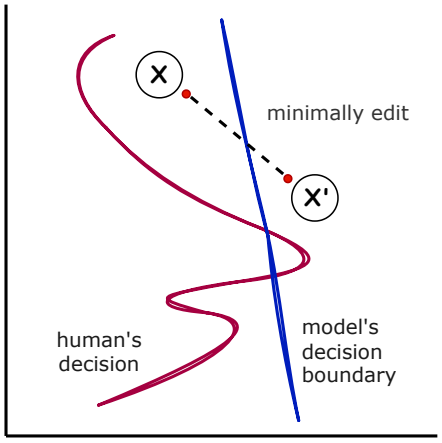

Figure 2: The counterfactual explanation and the original input are on different sides of the model's decision boundary, but on the same side of human's decision.

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.

3 Method

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

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 easily 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 the 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 the robustness of the model. In Step 3, we use both $\mathcal{O}$ and $\mathcal{A}$ to fine-tune f_θ.

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

3.1 Sampling unconfident instances

In Active Learning, several strategies are proposed to calculate the confidence score of classification models. Sampling strategies include various types of methods. Random sampling is the most common one that uses a random value as a confidence score. While it preserves the original distribution of the dataset, it does not guarantee that the chosen examples will be the most unconfident ones. Margin sampling calculates the difference between the top-2 prediction probabilities. Least-confidence sampling returns the ratio between the most confident prediction and 1 (100% confidence). Entropy-based sampling returns the entropy of predicted probability distribution $P(Y|x)$.

In our work, for each input $x \in \mathcal{O}$, we use the Entropy-based Sampling method to calculate the confidence score of a model. Let $P_\theta(Y|x)$ be a probability distribution over n classes predicted by f_θ (i.e., $Y \in \{1, 2, ..., n\}$). Entropy is calculated as follows:

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

3.2 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) (Garg and Ramakrishnan, 2020), 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. (2020) 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 example, they choose the ones that decrease the prediction probability of example the most.

In our work, we use BAE-R, 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.

3.2.1 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 BAE uses the average attention that the pre-trained language model gives to every token from all the layers.

3.2.2 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. Firstly, we replace an important token $w \in x$ with a semantically similar token. We replace the important token w with [MASK] and then use a masked language model g_ϕ (in our experiments, RoBERTa-large (Liu et al., 2019)) 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 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.

Secondly, we check if the perturbed sentence x' is a counterfactual explanation. Specifically, we check if 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 Step 1 and Step 2 with the second-most important token. We denote $\mathcal{A}$ as a set of generated counterfactual explanations.

3.3 Finetuning pre-trained Language model

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 fine-tune 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.

4 Evaluation

Our main hypothesis is that counterfactual explanations could help improve the robustness of the pre-trained models. Our evaluation aims to explore two different types of robustness in our expeirments:

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

[3] https://github.com/aesuli/SentiWordNet

[4] https://www.nltk.org/

Table 1: In-domain evaluation on the NLI and SA tasks.

Model	Accuracy
rb-mnli	90.01
rb-mnli w/ CAT	**91.44**
rb-tw	74.20
rb-tw w/ CAT	**77.15**

Table 2: In-domain evaluation on a subset of development set (Dev.) and their corresponding adversarial examples (Adv.).

Model	Accuracy	
	Dev.	Adv.
rb-mnli	**90.35**	9.65
rb-mnli w/ CAT	90.16	**15.12**
rb-tw	74.36	22.96
rb-tw w/ CAT	**75.57**	**41.92**

Do counterfactual explanations help improve the accuracy of NLP models on unseen data from (i) the *same domain* as the training data? and (ii) the *different domain* from 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

We evaluate our hypothesis on Natural Language Inference (NLI) and Sentiment Analysis (SA), two representative NLP tasks.

For NLI, we use the training dataset of MNLI (Williams et al., 2017) as $\mathcal{D}$. For in-domain evaluation, we randomly sample 550 instances from the validation dataset of MNLI (henceforth, $\mathcal{O}_{\text{dev}}$). For out-of-domain evaluation, we evaluate on NLI Diagnostics (Wang et al., 2018), HANS (McCoy et al., 2019), FEVER-NLI (Thorne et al., 2018), and ANLI (Nie et al., 2019).

For SA, we use the training dataset of TweetEval (Rosenthal et al., 2019) as $\mathcal{D}$. For in-domain evaluation, we randomly sample 750 instances from TweetEval development sets (henceforth, $\mathcal{O}_{\text{dev}}$) with the same distribution of each label. For out-of-domain evaluation, we evaluate on FinancialPhraseBank (Malo et al., 2014), IMDB (Maas et al., 2011), FiQA[5], StockTweet[6], Amazon (Keung et al., 2020), and Yelp (Zhang et al., 2015).

Table 3: Sensitivity to adversarial examples.

	% flipped ↓
rb-mnli	97.99
rb-mnli w/ CAT	**89.75**
rb-tw	96.53
rb-tw w/ CAT	**59.95**

For both tasks, we sample approximately 1,200 unconfident instances (i.e. $\mathcal{O}$) for CAT. We use accuracy as an evaluation metric.

4.2 Models

For the pre-trained model f_θ, we employed publicly available RoBERTa-large checkpoints finetuned on the training portion of MNLI corpus[7] (rb-mnli) and TweetEval dataset[8] (rb-tw).

For CAT, we do a pre-processing step by setting their maximum length to 512, then pad and truncate if they exceed the limitation setup. We use grid search to determine the learning rate. For NLI task, we setup three candidates, 1e-3, 1e-5, and 1e-8, and choose the one that achieves the best result overall. For SA task, we setup at 1e-3, 1e-5, and 1e-7.

4.3 Results and discussion

4.3.1 In-domain evaluation

Table 1 shows the result of in-domain evaluation. The results show that CAT improves the original models by 1.44% for the NLI task and by 2.95% for the SA task. This indicates that CAT is a promising approach to improving the robustness of NLP models to in-domain unseen inputs. Below, we give in-depth analyses of the behavior of CAT.

CAT-enhanced models precisely target adversarial examples without hurting the original performance. To analyze the behavior of CAT, we also evaluate our models on adversarial examples. We sample 750 and 550 instances from $\mathcal{O}_{\text{dev}}$ (for NLI and SA task respectively) and generated adver-

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

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

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

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

Table 4: Out-of-domain evaluation on the NLI task.

Model	ANLI	Diagnostics	FEVER	HANS
rb-mnli	31.8	66.39	70.7	73.13
rb-mnli w/ CAT	**32.37**	**66.49**	**70.87**	**73.75**

Table 5: Out-of-domain evaluation on the SA task.

Model	IMDB	FiQA	Stock	Yelp	FinP	Amazon
rb-tw	77.1	76.59	55.15	68.8	67.51	69.38
rb-tw w/ CAT	**78.24**	**78.37**	**57.84**	**69.31**	**69.17**	**69.98**

sarial examples by our counterfactual explanation method (§3.2). The results are shown in Table 2.

For the NLI task, the difference is not significant between rb-mnli and the CAT-enhanced rb-mnli (90.35% and 90.16%). However, for adversarial examples, the CAT-enhanced model outperforms rb-mnli by a large margin (5.47%).

For the SA task, we also compare rb-tw with the CAT-enhanced rb-tw. We found that the CAT-enhanced model is even over 1% higher on the development set. Besides, our CAT-enhanced model gains an impressive improvement on the adversarial examples, approximately 20% higher than the original model.

The results imply that our CAT-enhanced models precisely improve the robustness, while maintaining the in-domain performance.

CAT-enhanced models are more difficult to be fooled. We also investigate another in-domain robustness: given a sample $x \in O_{\text{dev}}$ and its prediction $f_\theta(x)$, we check if a model flips its prediction for its adversarial version x' generated by our counterfactual explanation method, i.e., $f_\theta(x) = f_\theta(x')$ or not. Ideally, the model should not flip the prediction, as x' is generated to maintain its gold label. The results are shown in Table 3.

Generally, our CAT-enhanced models achieve much lower flip rates than the original models. While the CAT-enhanced rb-mnli improves true prediction by more than 8%, the CAT-enhanced rb-tw decreases false prediction by around 36.5%.

The results reveal that CAT makes the model not easily fooled by new counterfactual explanations and more robust to them.

4.3.2 Out-of-domain evaluation

Tables 4 and 5 show the results of out-of-domain evaluation for NLI and SA tasks, respectively. For the NLI task, our CAT-enhanced models gain a minor improvement (roughly 1%) for all datasets compared to the original rb-mnli model. Our CAT-enhanced fine-tuned r-mnli achieves the best result on the HANS dataset and ANLI dataset. Besides, it insignificantly improves on the NLI Diagnostics and Fever NLI dataset.

For the SA task, CAT-enhanced rb-tw gets better performance (about 2%) for most out-of-domain datasets compared to the original model. Our CAT-enhanced rb-tw 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 improvement (0.6-1.14%).

Overall, the results indicate that CAT-enhanced models outperform the original models not only in the same domain but also in the different domain from training data.

5 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 hypothesis 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 CAT 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.

Acknowledgements

Special thanks to Tien Dang Huu for his comments and recommendations for our work.

References

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. 2021. Recent advances in adversarial training for adversarial robustness. *arXiv preprint arXiv:2102.01356*.

Brian Barr, Matthew R Harrington, Samuel Sharpe, and C Bayan Bruss. 2021. Counterfactual explanations via latent space projection and interpolation. *arXiv preprint arXiv:2112.00890*.

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.

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. *arXiv preprint arXiv:2105.03075*.

Siddhant Garg and Goutham Ramakrishnan. 2020. Bae: Bert-based adversarial examples for text classification. *CoRR*.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified robustness to adversarial word substitutions. *arXiv preprint arXiv:1909.00986*.

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.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A Smith. 2020. The multilingual amazon reviews corpus. *arXiv preprint arXiv:2010.02573*.

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. *arXiv preprint arXiv:1907.11692*.

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.

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.

R Thomas McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *arXiv preprint arXiv:1902.01007*.

Nafise Sadat Moosavi, Marcel de Boer, Prasetya Ajie Utama, and Iryna Gurevych. 2020. Improving robustness by augmenting training sentences with predicate-argument structures. *arXiv preprint arXiv:2010.12510*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

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

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.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2019. Semeval-2017 task 4: Sentiment analysis in twitter. *arXiv preprint arXiv:1912.00741*.

Alexis Ross, Ana Marasović, and Matthew E Peters. 2020. Explaining nlp models via minimal contrastive editing (mice). *CoRR*.

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.

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

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. *arXiv preprint arXiv:1803.05355*.

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. *arXiv preprint arXiv:1804.07461*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S Weld. 2021. Polyjuice: Generating counterfactuals for explaining, evaluating, and improving models. *arXiv preprint arXiv:2101.00288*.

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