AN INVESTIGATION OF THE (IN)EFFECTIVENESS OF COUNTERFACTUALLY AUGMENTED DATA

Nitish Joshi
Department of Computer Science
New York University
nitish@nyu.edu

He He
Department of Computer Science &
Center for Data Science
New York University
hehe@cs.nyu.edu

ABSTRACT

Numerous recent works show that current models exploit spurious correlations in benchmark datasets. Even though they achieve excellent in-distribution performance, they generalize poorly to out-of-distribution (OOD) data. Recent work has explored using counterfactually-augmented data (CAD)—data generated by minimally perturbing examples to flip the ground-truth label—to identify robust features. However, the OOD generalization results using CAD have been mixed (e.g. on natural language inference and question answering). To understand CAD better and explain this discrepancy, we draw insights from a simple linear regression model and demonstrate the potential pitfalls of CAD and how they are evident in current datasets. Specifically, we show that perturbations corresponding to one robust feature may not be useful for learning other robust features, and can hurt performance in worst-case scenarios. Our results suggest that CAD is limited by the specific robust features which are perturbed during crowdsourcing, and it is not immediately consequent that they would help OOD generalization.

1 INTRODUCTION

Large-scale datasets (Bowman et al., 2015; Rajpurkar et al., 2016) have enabled tremendous progress in Natural Language Processing (NLP) with the rise of pre-trained language models (Devlin et al., 2019; Peters et al., 2018). Despite the progress, there have been numerous works showing that models rely on spurious correlations in the datasets (McCoy et al., 2019; Naik et al., 2018; Tu et al., 2020; Wang & Culotta, 2020), i.e. correlations which are effective on a specific dataset but do not hold in general (e.g. high lexical overlap in SNLI dataset (Bowman et al., 2015) is spuriously correlated with entailment label).

A recent promising direction in building more robust models has been to collect counterfactually-augmented data (CAD) (Kaushik et al., 2020)—data generated by minimally perturbing examples from existing benchmark datasets to flip the ground-truth label. Figure 1 gives an example from natural language inference. By intervening on the robust features, the model is expected to learn to disentangle the spurious and non-spurious correlations.

Despite some recent work which tries to explain the efficacy of CAD by analyzing the underlying causal structure (Kaushik et al., 2021), there have been a recent body of work showing that CAD may not necessarily be helping with out-of-distribution (OOD) generalization. Specifically, Huang et al. (2020) show that CAD does not yield better generalization for Natural Language Inference (NLI). Similarly, Khashabi et al. (2020) find that for Question Answering (QA), unaugmented datasets give better performance when the annotation cost and datasets sizes are controlled.

In this work, we take a step towards bridging this gap between what theory suggests and what we observe in practice in regards to CAD. Firstly, we analyze a simple linear regression model trained on CAD and show that CAD corresponding to one robust feature can prevent the model from learning other robust features. Next, we empirically demonstrate how this issue is evident in existing datasets. Specifically, we categorize the perturbations present in CAD into different *perturbation types* (Wu et al., 2021) (eg. negating a sentence, or changing the numerals), and analyze how models

Premise: Several farmers bent over working on the fields while lady with a baby and four other children accompany them.
Original Hypothesis: The lady has three children. (Contradiction)
New Hypothesis: The lady has many children. (Entailment)

Figure 1: Illustration of a counterfactual example in Natural Language Inference where the numeral is modified to flip the label.

generalize to held-out perturbations. We find that model performance sometimes degrades on unseen perturbation types compared to models trained on unaugmented datasets.

Our results imply that CAD might be very specific to the robust features which are perturbed, and it is not immediately consequent that they are useful for OOD generalization. The results suggest that we might need more innovation on the crowdsourcing procedure for collecting counterfactual examples—we need to either collect targeted counterfactual examples (those which identify and fill the gaps of current models) or collect more diverse counterfactual examples.

2 Formulation and Analysis

In this section, we formalize the approach of counterfactual augmentation and discuss under what conditions it could be effective using a simple toy linear regression model for easier analysis.

2.1 Problem Statement

Consider data generated by the following process: A binary label $Y \in \{-1, 1\}$ is drawn with uniform probability ; Two features X_1 and X_2 follow Gaussian distributions conditioned on $Y = y$:

$$X_1 \mid y \sim \mathcal{N}(y\mu_1, \sigma_1^2) \,, \qquad X_2 \mid y \sim \mathcal{N}(y\mu_2, \sigma_2^2) \,. \tag{1}$$

We assume X_1 is a *robust feature* whose distribution does not change at test time, whereas X_2 is a *spurious feature* where μ_2 and σ_2 may change at test time, e.g. $\mu_2^{\text{train}} = 1$ and $\mu_2^{\text{test}} = -1$, thus relying on X_2 may lead to poor performance at test time. We aim to learn a linear model by least square regression: $Y = f(X) = w_1 X_1 + w_2 X_2$.

Without additional knowledge, X_2 appears to be a useful feature on the training set and will have non-zero weight. Specifically, we have

$$w_1 = \frac{\text{Cov}(X_1, Y)}{\text{Var}(X_1)} = \frac{\mu_1}{\mu_1^2 + \sigma_1^2} \,, \qquad w_2 = \frac{\text{Cov}(X_2, Y)}{\text{Var}(X_2)} = \frac{\mu_2}{\mu_2^2 + \sigma_2^2} \,. \tag{2}$$

2.2 Counterfactual Augmentation

In CAD, we edit a training example to flip its ground-truth label. Here we model the generation process by an *edit vector* $c \in \mathbb{R}^2$. Let X_i^c denote the feature value after editing. Intuitively, the counterfactual data centers around a shifted mean. Formally, we have

$$X_1^c \mid y \sim \mathcal{N}(-y(\mu_1 + c_1), \sigma_1^2) \,, \qquad X_2^c \mid y \sim \mathcal{N}(-y(\mu_2 + c_2), \sigma_2^2) \,. \tag{3}$$

Further, assuming that $\alpha\%$ of examples are edited to produce their counterfactuals, the distribution of the augmented data X^a, and the new learned weights for linear regression are:

$$X^a \mid y \sim \begin{cases} X \mid y & \text{w.p. } 1 - \alpha \\ X^c \mid y & \text{w.p. } \alpha \end{cases} \tag{4}$$

$$w_1 = \frac{(1 - 2\alpha)\mu_1 - c_1\alpha}{\sigma_1^2 + (\mu_1 + c_1)^2} \,, \qquad w_2 = \frac{(1 - 2\alpha)\mu_2 - c_2\alpha}{\sigma_2^2 + (\mu_2 + c_2)^2} \,. \tag{5}$$

Train Data	All types	quantifier	negation	lexical	insert	delete	resemantic
SNLI seed	$67.84_{0.84}$	$\mathbf{74.36}_{0.21}$	$\mathbf{69.25}_{2.09}$	$75.16_{0.32}$	$74.94_{1.05}$	$65.76_{2.34}$	$\mathbf{76.77}_{0.74}$
lexical	$70.44_{1.07}$	$72.42_{1.58}$	$68.75_{2.16}$	$\mathbf{81.81}_{0.99}$	$74.04_{1.04}$	$67.04_{3.00}$	$74.93_{1.16}$
insert	$66_{1.41}$	$68.15_{0.88}$	$57.75_{4.54}$	$71.08_{2.53}$	$\mathbf{78.98}_{1.58}$	$68.8_{2.71}$	$71.74_{1.53}$
resemantic	$\mathbf{70.8}_{1.68}$	$70.77_{1.04}$	$67.25_{2.05}$	$77.23_{2.35}$	$76.59_{1.12}$	$\mathbf{70.4}_{1.54}$	$75.40_{1.44}$

Table 1: Results for the different perturbation types in NLI (mean and std. deviation across 5 random seeds). We observe that models perform well on *aligned test sets* but do not perform well on *unaligned test sets*, sometimes doing worse than baseline.

2.3 Analysis

Let's consider an ideal edit vector in this toy setting: $c^* = [-2\mu_1, 0]$. This means that for each example, we keep the spurious feature x_2 the same, and move x_1 by $-2\mu_1$; the resulting counterfactual data will have maximum likelihood under our data generating distribution in (1). Further, let's assume that $\alpha = 0.5$, which means that every example is edited to produce its counterfactual. In this idea case, the weights are $w_1 = \frac{\mu_1}{\mu_1^2+\sigma_1^2}$ and $w_2 = 0$.

This shows that with proper c and α, counterfactual augmentation can effectively guard against spurious features (X_2 in this case). However, suppose both X_1 and X_2 are robust features, then not editing X_2 would set its weight to zero and potentially hurt the performance (e.g. if X_2 is less noisy than X_1 thus leads to better accuracy). In practice, this happens when there are multiple robust features but only a few are perturbed during counterfactual augmentation. For example, in the extreme case where all entailment examples are flipped to non-entailed ones by inserting a negation word, then the model will only rely on negation to make predictions.

3 Experimental Setup

Perturbation Types. Unlike the toy example, in NLP it is not always possible to define robust features using surface features such as text spans, since robust features typically involve the semantics of the text (e.g. negating the sentiment of a text) and the effect of a text span (e.g. negation words) largely depends on the context. We therefore define robust features as latent variables (e.g. sentiment, tense, action etc.) that generate the sentence form. Thus perturbing a robust feature would be reflected as a change in certain words in the sentence.

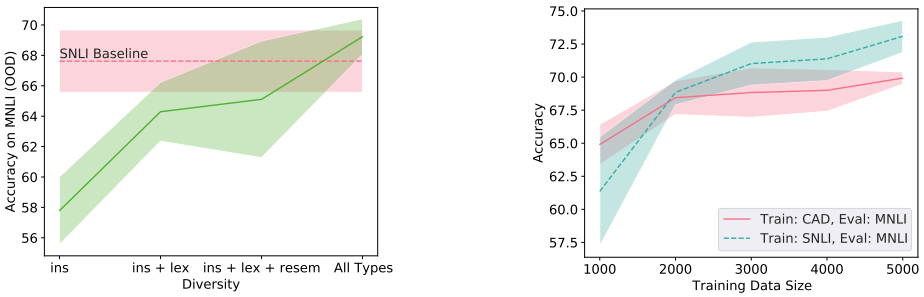
To uncover the latent robust features, we use simple linguistically-inspired rules (Wu et al., 2021) to roughly categorize edits into *perturbation types*: [1]: negation (change in negation modifier), quantifier (change in numeral POS tags), lexical (replace words without breaking POS tags), insert (only insert words/short phrases), delete (only delete words/short phrases) and resemantic (replace short phrases without affecting rest of the parsing tree). Note that all our train and test sets are paired i.e. contain the original example and its counterfactual.

Train/test sets. All our training sets contain paired CAD corresponding to a particular perturbation type, containing 700 seed examples and 700 perturbations. The baseline dataset (SNLI seed) contains SNLI examples of same total size. To test the generalization, we define two different types of test sets: *aligned test sets* are defined to be test sets which contain paired examples generated by the same perturbation type as in the training data (eg. training on examples from lexical perturbation type and testing on other examples from lexical perturbation types); and *unaligned test sets* which refer to the other test sets corresponding to perturbation types not seen in the training data.

4 Empirical Results

CAD performs well on aligned test sets - We see that on average models trained on the three categories perform very well on their *aligned test sets*, but do not always do well *unaligned test sets*.

[1] These types are not mutually exclusive, and we set a precedence order in case of ambiguity. We give more details in the Appendix. Considering the small dataset sizes, we use 3 types during training: lexical, insert and resemantic.



(a) (b)

Figure 2: Performance of models trained on CAD with increasing diversity with controlled dataset sizes — models generalize better OOD as the diversity increases. The figure on right shows that CAD is most beneficial in low data setting.

Note that resemantic as defined is a very broad category, and hence models trained on resemantic seem to perform well on other perturbation types.

CAD sometimes performs worse than baseline on unaligned test sets - Models trained on insert do much worse than the SNLI baseline on the lexical test set - this means that augmenting perturbations corresponding to one robust feature could also end up hurting other robust features. This is further corroborated by noting that the SNLI baseline performs best on both quantifier and negation, which means adding perturbations corresponding to any other types hurts these robust features.

4.1 Generalization to Out-of-Distribution Data

In the previous section, we have seen that training on CAD generated by a single perturbation type does not generalize well to unseen perturbation types. However, in practice CAD contains many different perturbation types. Do they cover enough robust features to enable OOD generalization?

Increasing Diversity: To test if increasing diversity of robust features leads to better OOD generalization, we create subsets of CAD with increasing number of perturbation types while keeping the total dataset size fixed. We also include an additional experiment ('All Types') where a model is trained on all perturbation types, with same size as the other datasets. We observe from Figure 2a that increasing diversity of perturbation types does indeed lead to better OOD generalization.

Role of Dataset Size: In Figure 2a, we observe 'All Types' performs better than baseline on OOD test set — this may indicate that CAD is indeed beneficial, in contrast with what was found by Huang et al. (2020). We argue that this effect is observed since we only train on a small amount of CAD dataset. To demonstrate this, we train models on increasing amounts of CAD, and corresponding amount of seed SNLI examples. The results are shown in Figure 2b. We see that CAD is beneficial for OOD generlization in very low data setting, and its benefits taper off with increasing data.[2]

5 Conclusion

In this work, we analyzed a toy linear regression model to better understand CAD. We identified a potential pitfall of this method, and demonstrated how it is evident in existing CAD. An important future direction for this work is addressing the issues - we need more innovation in the crowdsourcing procedure to collect either more diverse examples, or targeted counterfactual examples.

[2]We suspect this happens since with increasing dataset sizes, the baseline SNLI has more diverse data whereas CAD is limited by the specific perturbation types which were used to generate the data.

REFERENCES

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://www.aclweb.org/anthology/D15-1075.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *NAACL-HLT*, 2018.

William Huang, Haokun Liu, and Samuel R. Bowman. Counterfactually-augmented SNLI training data does not yield better generalization than unaugmented data. In *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pp. 82–87, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.insights-1.13. URL https://www.aclweb.org/anthology/2020.insights-1.13.

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

Divyansh Kaushik, Amrith Setlur, Eduard H Hovy, and Zachary Chase Lipton. Explaining the efficacy of counterfactually augmented data. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=HHiiQKWsOcV.

Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. More bang for your buck: Natural perturbation for robust question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 163–170, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.12. URL https://www.aclweb.org/anthology/2020.emnlp-main.12.

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://www.aclweb.org/anthology/P19-1334.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2340–2353, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1198.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://www.aclweb.org/anthology/N18-1202.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://www.aclweb.org/anthology/D16-1264.

E. Rosenfeld, P. Ravikumar, and A. Risteski. The risks of invariant risk minimization. *arXiv preprint arXiv:2010.05761*, 2020.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633, 2020. doi: 10.1162/tacl_a_00335. URL https://www.aclweb.org/anthology/2020.tacl-1.40.

Zhao Wang and A. Culotta. Identifying spurious correlations for robust text classification. *ArXiv*, abs/2010.02458, 2020.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

Tongshuang Wu, Marco Tulio Ribeiro, Jeffrey Heer, and Daniel S. Weld. Polyjuice: Automated, general-purpose counterfactual generation, 2021.

A Experimental Details

We implemented all our models using the HuggingFace implementation Wolf et al. (2019). For the experiments on the different perturbation types, models were trained for 10 epochs and the best checkpoint was selected according to performance on a development set consisting of all the perturbation types. This was done to ensure that all models are fairly evaluated and aren't biased towards a particular robust feature in the model selection procedure. This method thus assumes access to a small amount of examples across all the perturbation types for model selection — an assumption which can be easily violated in practice, biasing the models even more towards the robust features available during training.

B Perturbation Types

Type	Definition & Example	# of examples
negation	*Change in negation modifier* A dog is not fetching anything.	112
quantifier	*Change in words with numeral POS tags* The lady has ~~many~~ → three children.	272
lexical	*Replace few words without breaking the POS tags* The boy is ~~swimming~~ → running.	1162
insert	*Only insert words or short phrases* The tall man is digging the ground.	1017
delete	*Only delete word or short phrases* The ~~lazy~~ person just woke up.	293
resemantic	*Replaced short phrases without affecting parsing tree* The actor ~~saw~~ → had just met the director.	2901

Table 2: Definition of the control codes as well as the number of examples found in the NLI training data released by Kaushik et al. (2020). For the examples for each of the perturbation types, the deleted words are shown in red and the newly added words are shown in green.

The perturbation types we use in this paper correspond to the control codes used in Wu et al. (2021). Since they have not released their implementaion and clear definitions of what each type corresponds, we do not use some of the perturbation types (eg. restructure, which is subsumed in resemantic if it satisfies the definition in our case). We give the definitions of each of the types we used along with the numher of perturbations found in the NLI counterfactually-augmented dataset released by Kaushik et al. (2020) in Table 2. Also note that some of these perturbations could belong to multiple types — we use a precedence order to assign categories depending on which categories are more broad. Specifically, we used the following precedence order - negation > quantifier > lexical > insert = delete > resemantic.

C Extension of Linear Regression Analysis

The analysis in Section 2 focused on a toy example with only two variables. Here we extend the analysis to a more general case, where we have sets of robust and spurious features.

C.1 Learning with Spurious Correlation

We adopt the setting in Rosenfeld et al. (2020): each example consists of *robust features* $x_r \in \mathbb{R}^d$, whose joint distribution with the label is invariant during training and testing, and *spurious features* $x_s \in \mathbb{R}^d$ whose distribution varies at test time. Specifically, the label $y \in \{-1, 1\}$ has a uniform distribution, and both the robust and spurious features are drawn from Gaussian distributions: we consider $x = [x_r, x_s] \in \mathbb{R}^{2d}$ generated by the following process:

$$x_r \mid y \sim \mathcal{N}(y\mu_r, \sigma_r^2 I) \,, \qquad x_s \mid y \sim \mathcal{N}(y\mu_s, \sigma_s^2 I) \,. \tag{6}$$

The corresponding data distribution is denoted by $\mathcal{D}$. Note that μ_s and σ_s may change at test time, thus relying on x_s may lead to poor performance.

We consider the setting with infinite samples and learn a linear model by least square regression. The least square solution is given by

$$\hat{w} = \left[\Sigma_r^{-1}\mu_r, \Sigma_s^{-1}\mu_s\right] , \tag{7}$$

where Σ_r and Σ_s are covariance matrices of x_r and x_s, respectively. Note that this model relies on spurious features x_s which can vary at test time, thus it may have poor performance. A robust model that is invariant to spurious correlation would ignore x_s:

$$w_{\text{inv}} = \left[\Sigma_c^{-1}\mu_r, 0\right] . \tag{8}$$

We define the error of w to be the squared loss with respect to predictions given by the robust model w_{inv}:

$$\ell(w) = \mathbb{E}_{x\sim\mathcal{D}}\left[(w_{\text{inv}}^T x - w^T x)^2\right] . \tag{9}$$

It is then easy to show that $\ell(\hat{w}) = \mu_s^T \Sigma_s^{-1} \mu_s$.

C.2 Counterfactual Augmentation

The counterfactual data is generated by editing an example to flip its ground-truth label. We model the augmentation by a label-dependent *edit vector* $z = [yz_r, yz_s] \in \mathbb{R}^{2d}$ that translates x to change its label from y to $-y$. Note that x with different labels are translated in opposite directions. Let x^c denote the feature after editing and we have

$$x_r^c \mid y \sim \mathcal{N}(-y(\mu_r + z_r), \sigma_r^2 I) \,, \tag{10}$$

$$x_s^c \mid y \sim \mathcal{N}(-y(\mu_s + z_s), \sigma_s^2 I) \,. \tag{11}$$

The model is then trained on paired examples (x, x^c), whose distribution is denoted by $\mathcal{D}_c$.

Optimal edits. Ideally, the counterfactual data should de-correlate x_s and y, thus only changing robust features x_r, i.e. $z = [z_r, 0]$. For examples with original label y, we choose z_r that maximizes the log-likelihood of the flipped label:

$$\begin{aligned} z_r^* &= \arg\max_{z_r \in \mathbb{R}^d} \mathbb{E}_{x\sim\mathcal{D}} \log p(-y \mid x + [yz_r, 0]) \\ &= -2\mu_r \end{aligned} \tag{12}$$

Using the optimal edits, the model trained on $\mathcal{D}_c$ recovers the robust model w_{inv}, demonstrating the effectiveness of CAD.

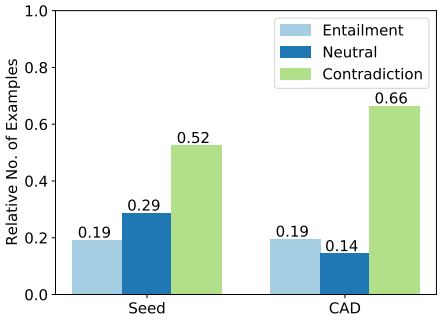


Figure 3: Fraction of entailment/neutral/contradiction examples in the SNLI seed set and CAD where the negation words are present in the hypothesis. We observe that the distribution is more skewed towards contradiction in CAD compared to the seed examples.

	Stress Test	**MNLI subset**
SNLI Seed	$\mathbf{57.51}_{4.63}$	$\mathbf{63.26}_{3.83}$
CAD	$49.58_{1.47}$	$55.66_{4.24}$

Table 3: Performance of models on stress test and the subset of MNLI, where both evaluate the extent to which models have learnt the negation bias. Models trained on CAD perform worse on both sets, implying that they exacerbate this spurious correlation.

Incomplete edits. There is an important assumption made in arriving at the above result: we have assumed *all* robust features are edited. Suppose we have two sets of robust features x_r and $x_{r'}$, then *not* editing $x_{r'}$ would effectively make it appear spurious to the model and indistinguishable from x_s. In practice, this happens when there are multiple robust features but only a few are perturbed during counterfactual augmentation, which is common during data collection since workers are instructed to edit a minimal number of words. For example, in the extreme case where all entailment examples are flipped to non-entailed ones by inserting a negation word, then the model will only rely on negation to make predictions.

More formally, using the same analysis, we can show that given $x = [x_r, x_{r'}, x_s]$ and incomplete edits $z = [z_r, 0, 0]$ that perturb only x_r (chosen by maximum likelihood estimation), the model learned on the augmented data is $\hat{w}_{\text{inc}} = [\Sigma_r^{-1}\mu_r, 0, 0]$ with an error $\ell(\hat{w}_{\text{inc}}) = \mu_{r'}^T \Sigma_{r'}^{-1} \mu_{r'}$. Assuming all variables have unit variance, then $\hat{w}_{\text{inc}}$ learned on the counterfactual data would have a larger error than $\hat{w}$ learned on the unaugmented data if $\|\mu_r'\|_2 > \|\mu_s\|_2$.

D CAD Exacerbates Existing Spurious Correlation

In this section, we analyze how CAD affects existing spurious correlations in the datasets. As an example, in the extreme case where all entailment examples are flipped to non-entailment by adding negation words (negation in Table 2), the model would learn to exclusively rely on negation words to make the prediction, which is clearly undesirable. We study the impact of CAD on two known spurious correlation in NLI datasets : word overlap bias McCoy et al. (2019); and negation word bias Gururangan et al. (2018).

Negation bias. We enumerate examples where there is a presence of a strong negation word (i.e. "no", "not", "n't") in the hypothesis. We plot the fraction of examples with a certain label in both the seed examples and the corresponding CAD examples in Figure 3. We observe that CAD contains a larger fraction of contradiction examples when negation words are present in the hypothesis.

To test if models trained on CAD suffer more from the negation bias, we train models on CAD and SNLI examples (of comparable size), and evaluate them on the 'negation' part of the stress test Naik et al. (2018). The stress test is a synthetically created challenge test, where the phrase "and false is not true" to appended to the hypothesis in the MNLI examples. In addition, we also evaluate on a

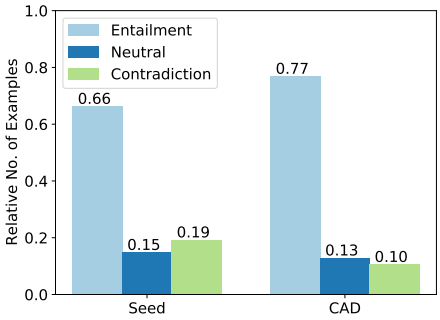


Figure 4: Fraction of entailment/neutral/contradiction examples in the SNLI seed set and CAD where the word-overlap bias is observed. The distribution is more skewed towards entailment in CAD than in the seed examples.

subset of the MNLI development set where the negation word is present in the hypothesis, and the gold label is not contradiction. The results are shown in Table 3. We observe that models trained on CAD perform worse on both of these test sets, implying that they rely more on the spurious correlation.

Word-overlap bias : We consider an NLI example to have high word overlap if more than 90% of words in the hypothesis are present in the premise. In Figure 4, we show the fraction of examples with a certain label in the high word overlap group from CAD and its SNLI seed examples, respectively. In both CAD and SNLI, the entailment examples are the majority in the high word overlap group, showing a strong correlation between high word overlap and entailment.[3] In fact, this correlation is stronger in CAD than SNLI, allowing models to rely on the word-overlap bias even more.

Next, we evaluate models trained on SNLI examples and CAD on the HANS challenge set McCoy et al. (2019), which specifically tests how much models rely on the word overlap bias. Unfortunately both models perform very poorly ($< 10\%$ accuracy) and hence it seems to be an unsuitable way to measure the extent to which models rely on this spurious correlation.

Takeaway: This section reveals that in the process of creating counterfactual examples, we may exacerbate existing spurious correlation. The fundamental challenge is that perturbations of the robust features are only observed through word change in the sentence—it is hard to surface the underlying causal variables without introducing artifacts on the sentence form.

[3]Note that the classes in both the seed examples and CAD are balanced, and hence this result cannot be attributed to class imbalance.