An Investigation of the (In)effectiveness of Counterfactually Augmented Data

Anonymous ACL submission

Abstract

While pretrained language models achieve excellent performance on natural language understanding benchmarks, they tend to rely on spurious correlations and 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 that are invariant under distribution shift. However, empirical results using CAD for OOD generalization have been mixed. To explain this discrepancy, we draw insights from a linear Gaussian model and demonstrate the pitfalls of CAD. Specifically, we show that (a) while CAD is effective at identifying robust features, it may prevent the model from learning *unperturbed* robust features; and (b) CAD may exacerbate existing spurious correlations in the data. On two crowdsourced CAD datasets, our results show that the lack of perturbation diversity limits their effectiveness on OOD generalization, calling for innovative crowdsourcing procedures to elicit diverse perturbation of examples.

1 Introduction

Large-scale datasets have enabled tremendous progress in natural language understanding (NLU) (Rajpurkar et al., 2016; Wang et al., 2018a) with the rise of pretrained 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, i.e. heuristics that are effective on a specific dataset but do not hold in general (McCoy et al., 2019; Naik et al., 2018; Wang and Culotta, 2020). For example, high word overlap is found to be associated with entailment in natural language inference (NLI) datasets.

A recent promising direction is to collect counterfactually-augmented data (CAD) (Kaushik et al., 2020) by asking humans to minimally edit examples to flip their ground-truth label.[1] Figure 1 shows example edits for NLI. Given interventions on *robust features* that "cause" the label to change, the model is expected to learn to disentangle the spurious and robust features.

Despite recent attempt to explain the efficacy of CAD by analyzing the underlying causal structure of the data (Kaushik et al., 2021), empirical results on out-of-distribution (OOD) generalization using CAD are mixed. Specifically, Huang et al. (2020) show that CAD does not improve OOD generalization for NLI; Khashabi et al. (2020) find that for question answering, unaugmented datasets give better performance when the annotation cost and dataset size 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. An intuitive example to illustrate our key observation is shown in Figure 1 (a), where the verb 'eating' is changed to 'drinking' to flip the label. While there are many other words that could have been changed to flip the label, from this pair of examples the model learns to use *only* the verbs (e.g. using a Naive Bayes model, all other words would have zero weights). As a result, this model would fail when evaluated on examples such as those in (b) where the quantifier 'two' is changed to 'three', while a model trained on the unaugmented data may learn to use the quantifiers.

Concretely, we formalize counterfactual augmentation using a linear Gaussian model and show that perturbations of one robust feature can prevent the model from learning other robust features. We then empirically demonstrate this issue in two CAD datasets collected for NLI and Question Answering (QA) . We identify the robust features by categoriz-

[1] Throughout the rest of the paper, CAD refers to counterfactually-augmented data containing pairs of the original example and a corresponding revised example.

Premise: The lady is standing next to her two children who are eating a pizza.
Original Hypothesis: The two children near the lady are ~~eating~~ something. (Entailment)
Revised Hypothesis: The two children near the lady are drinking something. (Contradiction)

(a)

Premise: The lady is standing next to her two children who are eating a pizza.
Original Hypothesis: The ~~two~~ children near the lady are eating something. (Entailment)
Revised Hypothesis: The three children near the lady are eating something. (Contradiction)

(b)

Figure 1: Illustration of counterfactual examples in Natural Language Inference. Augmenting examples like (a) hurts performance on examples like (b) where a different robust feature has been perturbed, since the first example encourages the model to exclusively focus on the highlighted words.

ing the edits into different *perturbation types* (Wu et al., 2021) (e.g. negating a sentence or changing the quantifiers), and show that models do not generalize well to unseen perturbation types, sometimes even performing worse than models trained on unaugmented data.

Our analysis of the relation between perturbation types and generalization can help explain other observations such as CAD being more beneficial in the low-data regime. With increasing data size, improvement from using CAD plateaus compared to unaugmented data, suggesting that the number of perturbation types in existing CAD datasets does not keep increasing.

.

Another consequence of the lack of diversity in edits is annotation bias, which may produce spurious correlations similar to what happened in standard crowdsourcing procedures. While CAD is intended to debias the dataset, surprisingly, we find that crowdsourced CAD for NLI exacerbates word overlap bias (McCoy et al., 2019) and negation bias (Gururangan et al., 2018a) observed in existing benchmarks.

In sum, we show that the effectiveness of current CAD datasets is limited by the set of robust features that are perturbed. Furthermore, they may exacerbate spurious correlations in existing benchmarks. Our results highlight the importance of increasing the diversity of counterfactual perturbations during crowdsourcing: We need to elicit more diverse edits of examples and collect targeted counterfactual examples that fix bugs in current models.

2 Analysis of a Linear Model

In this section, we formalize counterfactual augmentation and discuss under what conditions it is effective using a linear Gaussian model and squared loss.

2.1 Learning under Spurious Correlation

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

$$y = \begin{cases} 1 & \text{w.p. } 0.5 \\ -1 & \text{otherwise} \end{cases} \quad (1)$$

$$x_r \mid y \sim \mathcal{N}(y\mu_r, \sigma_r^2 I) \ , \quad (2)$$

$$x_s \mid y \sim \mathcal{N}(y\mu_s, \sigma_s^2 I) \ , \quad (3)$$

where $\mu_r \in \mathbb{R}^{d_r}$; $\mu_s \in \mathbb{R}^{d_s}$; $\sigma_r, \sigma_s \in \mathbb{R}$; and I is the identity matrix.[2] The corresponding data distribution is denoted by $\mathcal{D}$. Note that the relation between y and the spurious features x_s depends on μ_s and σ_s, which may change at test time, thus relying on x_s may lead to poor OOD performance.

We consider the setting with infinite samples and learn a linear model ($y = w^T x$ where $w \in \mathbb{R}^d$) by least square regression. Let $\hat{w} \in \mathbb{R}^d$ be the optimal solution on $\mathcal{D}$ (without any counterfactual augmentation). The closed form solution is:

$$\mathrm{Cov}(x, x)\hat{w} = \mathrm{Cov}(x, y) \quad (4)$$

$$\hat{w} = \mathrm{Cov}(x, x)^{-1}\mu \quad (5)$$

[2]This model corresponds to the anti-causal setting (Scholkopf et al., 2012), i.e. the label causing the features. We adopt this setting since it is consistent with how most data is generated in tasks like NLI, sentiment analysis etc.

where $\mu = [\mu_r, \mu_s] \in \mathbb{R}^d$ and $\text{Cov}(\cdot)$ denotes the covariance matrix:

$$\text{Cov}(x, x) = \begin{bmatrix} \Sigma_r & \mu_r \mu_s^T \\ \mu_s \mu_r^T & \Sigma_s \end{bmatrix}, \quad (6)$$

where Σ_r, Σ_s are covariance matrices of x_r and x_s respectively. This model relies on μ_s that can vary at test time, thus it may have poor performance under distribution shift. A robust model w_{inv} that is invariant to spurious correlations would ignore x_s:

$$w_{\text{inv}} = \left[\Sigma_r^{-1}\mu_r, 0\right]. \quad (7)$$

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

$$\ell(w) = \mathbb{E}_{x\sim\mathcal{D}}\left[(w_{\text{inv}}^T x - w^T x)^2\right]. \quad (8)$$

2.2 Counterfactual Augmentation

The counterfactual data is generated by editing an example to flip its label. We model the perturbation by an *edit vector* z that translates x to change its label from y to $-y$ (i.e. from 1 to -1 or vice versa). For instance, the counterfactual example of a positive example $(x, +1)$ is $(x + z, -1)$. Specifically, we define the edit vector to be $z = [yz_r, yz_s] \in \mathbb{R}^d$, where $z_r \in \mathbb{R}^{d_r}$ and $z_s \in \mathbb{R}^{d_s}$ are the displacements for the robust and spurious features. Here, z is label-dependent so that examples with different labels are translated in opposite directions. Therefore, the counterfactual example $(x^c, -y)$ generated from (x, y) has the following distribution:

$$x_r^c \mid -y \sim \mathcal{N}(y(\mu_r + z_r), \sigma_r^2 I), \quad (9)$$

$$x_s^c \mid -y \sim \mathcal{N}(y(\mu_s + z_s), \sigma_s^2 I). \quad (10)$$

The model is then trained on the combined set of original examples x and counterfactual examples x^c, whose distribution is denoted by $\mathcal{D}_c$.

Optimal edits. Ideally, the counterfactual data should de-correlate x_s and y, thus it should only perturb the robust features x_r, i.e. $z = [yz_r, 0]$. To find the displacement z_r that move x across the decision boundary, we maximize the log-likelihood of the flipped label under the data generating distribution $\mathcal{D}$:

$$z_r^* = \arg\max_{z_r \in \mathbb{R}^{d_r}} \mathbb{E}_{(x,y)\sim\mathcal{D}} \log p(-y \mid x + [yz_r, 0])$$
$$= -2\mu_r. \quad (11)$$

Intuitively, it moves the examples towards the mean of the opposite class along coordinates of the robust features.

Using the edits specified above, if the model trained on $\mathcal{D}_c$ has optimal solution $\hat{w}_c$, we have:

$$\text{Cov}(x, x)\hat{w}_c = \text{Cov}(x, y)$$
$$\hat{w}_c = \left[\Sigma_r^{-1}\mu_r, 0\right] = w_{\text{inv}}. \quad (12)$$

Thus, the optimal edits recover the robust model w_{inv}, demonstrating the effectiveness of CAD.

Incomplete edits. There is an important assumption made in the above result: we have assumed *all* robust features are edited. Suppose we have two sets of robust features x_{r1} and x_{r2},[3] then *not* editing x_{r2} 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 can be common during data collection if workers rely on simple patterns to make the minimal edits). Considering the NLI example, if all entailment examples are flipped to non-entailment ones by inserting a negation word, then the model will only rely on negation to make predictions.

More formally, consider the case where the original examples $x = [x_{r1}, x_{r2}, x_s]$ and counterfactual examples are generated by incomplete edits $z = [z_{r1}, 0, 0]$ that perturb only x_{r1}. Using the same analysis above where z_{r1} is chosen by maximum likelihood estimation, let the model learned on the incompletely augmented data be denoted by $\hat{w}_{\text{inc}}$. We then have the following:

Proposition 1. *Assuming all variables have unit variance and* $\|\mu_r\| = 1$, $\ell(\hat{w}_{inc}) > \ell(\hat{w})$ *if* $\|\mu_{r2}\| > \|\mu_s\|$, *where* $\|\cdot\|$ *denotes the Euclidean norm.*

Proof Sketch. The proof mainly follows from algebra and using the fact that $\text{Cov}(x, x)^{-1}$ is a block matrix consisting of rank-one perturbations of the identity matrix. We refer the reader to Appendix A for the detailed proof. □

This shows that the error is more in the case of incomplete edits compared to the unaugmented case. Next, we show that the problem of incomplete edits is exhibited in real CAD too.

[3] We assume they are conditionally independent given the label.

Type	Definition	Example	# examples (NLI/QA)
`negation`	*Change in negation modifier*	A dog is not fetching anything.	200/683
`quantifier`	*Change in words with numeral POS tags*	The lady has ~~many~~ → three children.	344/414
`lexical`	*Replace few words without breaking the POS tags*	The boy is ~~swimming~~ → running.	1568/1737
`insert`	*Only insert words or short phrases*	The tall man is digging the ground.	1462/536
`delete`	*Only delete word or short phrases*	The ~~lazy~~ person just woke up.	562/44
`resemantic`	*Replaced short phrases without affecting parsing tree*	The actor ~~saw~~ → had just met the director.	2760/1866

Table 1: Definition of the perturbation types and the corresponding number of examples in the NLI CAD dataset released by (Kaushik et al., 2020) and the BoolQ CAD dataset released by Khashabi et al. (2020). In the example edits, the deleted words are shown in ~~red~~ and the newly added words are shown in green.

3 Diversity and Generalization in CAD

In this section, we test the following hypothesis based on the above analysis: models trained on CAD are limited to the specific robust features that are perturbed and may not learn other unperturbed robust features. We empirically analyze how augmenting counterfactual examples by perturbing one robust feature affects the performance on examples generated by perturbing other robust features.

3.1 Experiment Design

Perturbation types. Unlike the Gaussian example, in NLU it is not easy to define robust features since they typically correspond to the semantics of the text (e.g. sentiment). We therefore define robust features as latent variables that generate the sentence form (e.g. sentiment, tense, action). Perturbing a robust feature would change certain words in the sentence. As an example, in Figure 1 (b), perturbation of the quantity is reflected as a change in the word 'two' to 'three'. To uncover the latent robust features, we use linguistically-inspired rules (Wu et al., 2021) to categorize the edits into several *perturbation types*: `negation`, `quantifier`, `lexical`, `insert`, `delete` and `resemantic`. Table 1 gives the definitions of each type.[4]

Train/test sets. Both the training sets and the test sets contain counterfactual examples generated by a particular perturbation type. To test the generalization from one perturbation type to another, we use two types of test sets: *aligned test sets* where examples are generated by the same perturbation type as the training data; and *unaligned test sets* where examples are generated by unseen perturbation types (e.g. training on examples from `lexical` and testing on `negation`).

[4]Since these types are not mutually exclusive, we set a precedence order among them when there are ambiguities.

3.2 Experimental Setup

Data. We experiment on two CAD datasets collected for SNLI (Kaushik et al., 2020) and BoolQ (Khashabi et al., 2020). The size of the paired data (seed examples and edited examples) for each perturbation type is given in Table 1. Since some types (e.g. `delete`) contain too few examples for training, we train on the top three largest perturbation types: `lexical`, `insert`, and `resemantic` for SNLI; and `lexical`, `negation`, and `resemantic` for BoolQ.

For SNLI, to control for dataset sizes across all experiments, we use 700 seed examples and their corresponding 700 perturbations for each perturbation type. As a baseline ('SNLI seed'), we subsample examples from SNLI to create a similar sized dataset for comparison.[5]

For BoolQ (Clark et al., 2019a), our initial experiments show that training on only CAD does not reach above random-guessing. Thus, we include all original training examples in BoolQ (Khashabi et al., 2020), and replace part of them with CAD for each perturbation type. This results in a training set of 9427 examples of which 683 are CAD for each perturbation type. The size 683 is chosen to match the the smallest CAD type for BoolQ. As a baseline ('BoolQ seed'), we train on the complete BoolQ training set.

Model. We use the Hugging Face implementation (Wolf et al., 2019) of RoBERTa (Liu et al., 2019) to fine-tune all our models. To account for the small dataset sizes, we run all our experiments with 5 different random seeds and report the mean and standard deviation. Details on hyperparameter tuning are reported in Appendix B.1.

[5]We observe similar trends when using different subsets of the SNLI data. We report the mean and standard deviation across different subsets in Appendix B.3.

Train Data	lexical	insert	resemantic	quantifier	negation	delete
SNLI seed	$75.16_{0.32}$	$74.94_{1.05}$	$\mathbf{76.77}_{0.74}$	$\mathbf{74.36}_{0.21}$	$\mathbf{69.25}_{2.09}$	$65.76_{2.34}$
lexical	$\mathbf{79.70}_{2.07}$	$68.61_{5.26}$	$71.46_{3.07}$	$69.90_{3.83}$	$66.00_{2.99}$	$61.76_{5.27}$
insert	$67.83_{3.96}$	$\mathbf{79.30}_{0.39}$	$70.53_{2.19}$	$66.31_{3.10}$	$55.0_{4.10}$	$69.75_{2.43}$
resemantic	$77.14_{2.12}$	$76.43_{1.05}$	$75.31_{1.10}$	$71.26_{0.36}$	$66.75_{1.69}$	$\mathbf{70.16}_{1.09}$

Table 2: Accuracy of NLI CAD on both aligned and unaligned test sets. We report the mean and standard deviation across 5 random seeds. Each dataset has a total of 1400 examples. On average models perform worse on unaligned test sets (i.e. unseen perturbation types).

Train Data	lexical	negation	resemantic	quantifier	insert
BoolQ seed	$65.79_{2.11}$	$62.61_{2.65}$	$68.97_{1.83}$	$61.00_{1.65}$	$57.11_{0.67}$
lexical	$\mathbf{77.38}_{1.04}$	$64.32_{2.18}$	$\mathbf{80.78}_{1.46}$	$\mathbf{70.75}_{2.03}$	$\mathbf{66.77}_{1.35}$
negation	$63.18_{1.46}$	$\mathbf{72.91}_{2.31}$	$66.74_{2.22}$	$61.75_{2.44}$	$65.42_{1.45}$
resemantic	$72.29_{0.72}$	$64.92_{1.56}$	$75.60_{2.11}$	$70.00_{2.85}$	$64.91_{2.31}$

Table 3: Accuracy of BoolQ CAD on both aligned and unaligned test sets. We report the mean and standard deviation across 5 random seeds. Each dataset has a total of 9427 examples. On average models perform worse on unaligned test sets (i.e. unseen perturbation types).

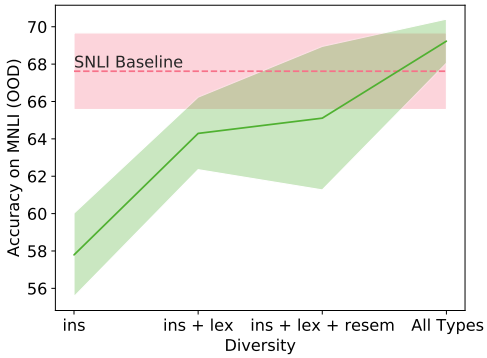


Figure 2: OOD accuracy on MNLI of models trained on SNLI CAD and SNLI seed (baseline) with increasing number of perturbation types and fixed training set size. More perturbation types in the training data leads to higher OOD accuracy.

3.3 Generalization to Unseen Perturbation Types

We discuss results for the main question in this section—how does adding CAD generated from one perturbation type affect performance on examples generated from other perturbation types? Table 2 and 3 show results for SNLI and BoolQ .

CAD performs well on aligned test sets. We see that on average models perform very well on the aligned test sets (same perturbation type as the training set), but do not always do well on unaligned test sets (unseen perturbation types), which is consistent with our analysis in Section 2. On SNLI, one exception is `resemantic`, which performs well on unseen perturbation types. We believe this is because it is a broad category (replacing any constituent) that covers other types such as `lexical` (replacing any word). Similarly, on BoolQ, `lexical` and `resemantic` both perform better than the baseline on some unaligned test sets (e.g. `quantifier`), but they perform much better on the aligned test sets.

CAD sometimes performs worse than the baseline on unaligned test sets. For example, on SNLI, training on `insert` does much worse than the seed baseline on `lexical` and `resemantic`, and SNLI seed performs best on `quantifier` and `negation`. On BoolQ, training on `negation` does slightly worse than the baseline on `lexical` and `resemantic`. This suggests that augmenting perturbations of one robust feature may prevent the model from learning other robust features (that could have been learned without the augmentation).

3.4 Generalization to Out-of-Distribution Data

In Section 3.3, 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. We first verify that increasing the number of perturbed robust features leads to better OOD generalization. Specifically, we train models on subsets of SNLI CAD with increasing coverage of perturbation types and eval-

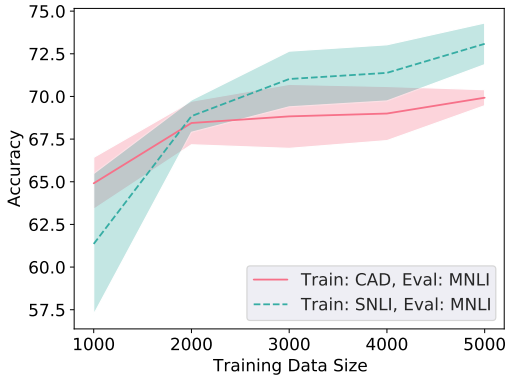


Figure 3: Accuracy on the OOD set (MNLI) for models trained on increasing amounts of NLI CAD. CAD is more beneficial in the low data regime, but its benefits taper off (compared to SNLI baseline of same size) as the dataset size increases.

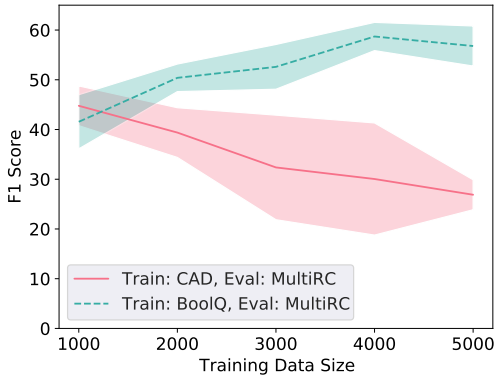


Figure 4: F1 score on the OOD set (MultiRC) for models trained on increasing amounts of QA CAD. CAD performs comparable to the baseline in the low data regime, but surprisingly performs worse with increasing dataset sizes, probably due to overfitting to a few perturbation types.

	BERT	RoBERTa
SNLI seed	$59.74_{0.29}$	$\mathbf{73.77}_{1.16}$
CAD	$\mathbf{60.18}_{1.05}$	$70.05_{1.15}$

Table 4: Accuracy (mean and std. deviation across 5 runs) on MNLI of different pretrained models fine-tuned on SNLI seed and CAD. CAD seems to be less beneficial when using better pretrained models.

uate on MNLI as the OOD data. Starting with only `insert`, we add one perturbation type at a time until all types are included; the total number of examples are fixed throughout the process at 1400 (which includes 700 seed examples and the corresponding 700 perturbations).

Figure 2 shows the OOD accuracy on MNLI when trained on CAD and SNLI seed examples of the same size. We observe that as the number of perturbation types increases, models generalize better to OOD data despite fixed training data size. The result highlights the importance of collecting a diverse set of counterfactual examples, even if each perturbation type is present in a small amount.

A natural question to ask here is: If we continue to collect more counterfactual data, does it cover more perturbation types and hence lead to better OOD generalization? Thus we investigate the impact of training data size next.[6]

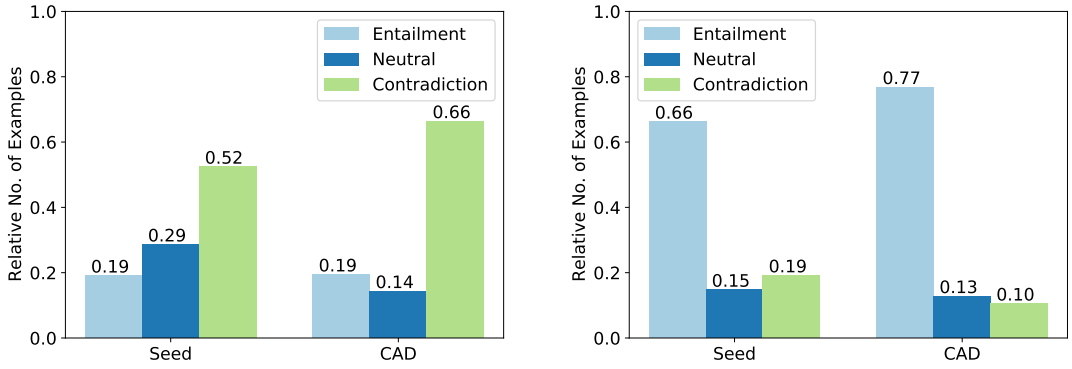
[6]The results in Figure 2 when all perturbation types are included indicate that CAD performs better than the SNLI baseline. This is not in contradiction with the results found in Huang et al. (2020), since our models are trained on only a subset of CAD. This further motivates the study of how CAD data size affects generalization.

Role of Dataset Size. To better understand the role dataset size plays in OOD generalization, we plot the learning curve on SNLI CAD in Figure 3, where we gradually increase the amount of CAD for training. The baseline model is trained on SNLI seed examples of the same size, and all models are evaluated on MNLI (as the OOD dataset). We also conduct a similar experiment on BoolQ in Figure 4, where a subset of MultiRC (Khashabi et al., 2018) is used as the OOD dataset following Khashabi et al. (2020). Since the test set is unbalanced, we report F1 scores instead of accuracy in this case.

For SNLI, CAD is beneficial for OOD generalization only in low data settings ($<$ 2000 examples). As the amount of data increases, the comparable SNLI baseline performs better and surpasses the performance of CAD. Similarly for BoolQ, we observe that CAD is comparable to the baseline in the low data setting ($\sim$ 1000 examples). Surprisingly, more CAD for BoolQ leads to worse OOD performance. We suspect this is due to overfitting to the specific perturbation types present in BoolQ CAD.

Intuitively, as we increase the amount of data, the diversity of robust features covered by the seed examples also increases. On the other hand, the benefit of CAD is restricted to the *perturbed* robust features. The plateaued performance of CAD (in the case of NLI) shows that the diversity of perturbations may not increase with the data size as fast as we would like, calling for better crowdsourcing protocols to elicit diverse edits from workers.

Role of Pretraining. Tu et al. (2020) show that larger pretrained models generalize better from mi-



(a) Negation bias

(b) Word-overlap bias

Figure 5: Fraction of entailment/neutral/contradiction examples in the SNLI seed set and CAD where (a) negation words are present in the hypothesis; (b) word overlap bias is observed. We observe that the distribution is more skewed in CAD compared to the seed examples (towards contradiction for the negation bias ((a)) and towards entailment for the word overlap bias ((b))

nority examples. Therefore, in our case we would expect CAD to have limited benefit on larger pretrained models since they can already leverage the diverse (but scarce) robust features revealed by SNLI examples. We compare the results of BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) trained on SNLI CAD in Table 4. For the RoBERTa model (pretrained on more data), CAD no longer improves over the SNLI baseline, suggesting that current CAD datasets may not have much better coverage of robust features than what stronger pretrained models can already learn from benchmarks like SNLI.

4 CAD Exacerbates Existing Spurious Correlation

An artifact of underdiverse perturbations is the newly introduced spurious correlations. As an example, in the extreme case where all entailment examples are flipped to non-entailment by the `negation` operation in Table 1, the model would learn to exclusively rely on the existence of negation words to make predictions, which is clearly undesirable. In this section, we study the impact of CAD on two known spurious correlations in NLI benchmarks: word overlap bias (McCoy et al., 2019) and negation bias (Gururangan et al., 2018b).

Negation bias. We take examples where there is a presence of a negation word (i.e. "no", "not", "n't") in the hypothesis, and plot the fraction of examples in each class in both the seed and the corresponding CAD examples in Figure 5a. As expected, contradiction is the majority class in the seed group, but surprisingly, including CAD amplifies the fraction of contradiction examples! As a result, training on CAD leads to worse performance on challenge sets that counter the negation bias compared to training on seed examples of the same size. Specifically, we test on the 'negation' part of the Stress Tests (Naik et al., 2018)[7] and challenge examples in the combined MNLI development set which contain negation words in the hypothesis but are not contradictions. Table 5 shows that models trained on CAD perform worse on both test sets, implying that they rely more on the negation bias.

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

Table 5: Accuracy of models on challenge examples in the stress test and MNLI, where non-contradiction examples contain a negation word in the hypothesis. Models trained on CAD perform worse on both sets, implying that they exacerbate the negation bias.

Word-overlap bias. Similarly, in Figure 5b, we show that CAD amplifies the fraction of entailment examples among those with high word overlap (i.e. more than 90% of words in the hypothesis are present in the premise). Models trained on SNLI and CAD both perform poorly (< 10% accuracy) on the non-entailment subset of HANS challenge set (McCoy et al., 2019), which exploits the word overlap bias.

Takeaway. This section reveals that in the process of creating CAD, we may inadvertently exacer-

[7] Synthetic examples where the phrase "and false is not true" is appended to the hypothesis of MNLI examples.

bate existing spurious correlations. The fundamental challenge here 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 (additional) artifacts to the sentence form.

5 Related Work

Label-Preserving Data Augmentation. A common strategy to build more robust models is to augment existing datasets with examples similar to those from the target distribution. Min et al. (2020) improve accuracy on HANS challenge set(McCoy et al., 2019) by augmenting syntactically-rich examples. Jia and Liang (2016); Andreas (2020) recombine examples to achieve better compositional generalization. There has also been a recent body of work using task-agnostic data augmentation by paraphrasing (Wei and Zou, 2019), back-translation (Sennrich et al., 2016) and masked language models (Ng et al., 2020). The main difference between these works and CAD, is that the edits in these works are label-preserving whereas they are label-flipping in CAD—the former prevents models from being over-sensitive and the latter alleviate under-sensitivity to perturbations.

Label-Changing Data Augmentation. Lu et al. (2020); Zmigrod et al. (2019) use rule-based CAD to mitigate gender stereotypes. Gardner et al. (2020) build similar contrast sets using expert edits for evaluation. In contrast, Kaushik et al. (2020) crowdsource minimal edits. Recently, Teney et al. (2020) also use CAD along with additional auxiliary training objectives and demonstrate improved OOD generalization.

Kaushik et al. (2021) analyze a similar toy model (linear Gaussian model) demonstrating the benefits of CAD, and showed that noising the edited spans hurts performance more than other spans. Our analysis complements theirs by showing that while spans identified by CAD are useful, a lack of diversity in these spans limit the effectiveness of CAD, thus better coverage of robust features could potentially lead to better OOD generalization.

Robust Learning Algorithms. Another direction of work has explored learning more robust models without using additional augmented data. These methods essentially rely on learning debiased representations—Wang et al. (2018b) create a biased classifier and project its representation out of the model's representation. Along similar lines, Belinkov et al. (2019) remove hypothesis-only bias in NLI models by adversarial training. He et al. (2019) and Clark et al. (2019b) correct the conditional distribution given a biased model. Utama et al. (2020) build on this to remove 'unknown' biases, assuming that a weak model learns a biased representations. More recently, Veitch et al. (2021) use ideas from causality to learn invariant predictors from counterfactual examples. The main difference between these methods and CAD, is that the former generally requires some prior knowledge of what spurious correlations models learn (e.g. by constructing a biased model or weak model), whereas CAD is a more general human-in-the-loop method that leverages humans' knowledge of robust features.

6 Conclusion and Future Directions

In this work, we first analyzed CAD theoretically using a linear model and showed that models do not generalize to unperturbed robust features. We then empirically demonstrated this issue in two CAD datasets, where models do not generalize well to unseen perturbation types. We also showed that CAD amplifies existing spurious correlations, pointing out another concern. Given these results, a natural question is: How can we fix these problems and make CAD more useful for OOD generalization? We discuss a few directions which we think could be helpful:

- We can use generative models (Raffel et al., 2020; Lewis et al., 2019) to generate *diverse* minimal perturbations and then crowdsource labels for them (Wu et al., 2021). We can improve the diversity of the generations by masking different spans in the text to be infilled, thus covering more robust features.

- An alternative to improving the crowdsourcing procedure, is to devise better learning algorithms which mitigate the issues pointed in this work. For example, given that we know the models do not always generalize well to unperturbed features, we can regularize the model to limit the reliance on the perturbed features.

We hope that this analysis spurs future work on CAD, making them more useful for OOD generalization.

References

Jacob Andreas. 2020. Good-enough compositional data augmentation. In *ACL*.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019a. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019b. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *EMNLP/IJCNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

M. Gardner, Y. Artzi, V. Basmova, J. Berant, B. Bogin, S. Chen, P. Dasigi, D. Dua, Y. Elazar, A. Gottumukkala, N. Gupta, H. Hajishirzi, G. Ilharco, D. Khashabi, K. Lin, J. Liu, N. F. Liu, P. Mulcaire, Q. Ning, S. Singh, N. A. Smith, S. Subramanian, R. Tsarfaty, E. Wallace, A. Zhang, and B. Zhou. 2020. Evaluating NLP models via contrast sets. In *Empirical Methods in Natural Language Processing (EMNLP)*.

S. Gururangan, S. Swayamdipta, O. Levy, R. Schwartz, S. R. Bowman, and N. A. Smith. 2018a. Annotation artifacts in natural language inference data. In *North American Association for Computational Linguistics (NAACL)*.

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

H. He, S. Zha, and H. Wang. 2019. Unlearn dataset bias for natural language inference by fitting the residual. In *Proceedings of the EMNLP Workshop on Deep Learning for Low-Resource NLP*.

William Huang, Haokun Liu, and Samuel R. Bowman. 2020. 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*, pages 82–87, Online. Association for Computational Linguistics.

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. *ArXiv*, abs/1606.03622.

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

Divyansh Kaushik, Amrith Setlur, Eduard H Hovy, and Zachary Chase Lipton. 2021. Explaining the efficacy of counterfactually augmented data. In *International Conference on Learning Representations*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface:a challenge set for reading comprehension over multiple sentences. In *NAACL*.

Daniel Khashabi, Tushar Khot, and Ashish Sabharwal. 2020. 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)*, pages 163–170, Online. Association for Computational Linguistics.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and A. Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. 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*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353,

Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. 2020. SSMBA: Self-supervised manifold based data augmentation for improving out-of-domain robustness. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. 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)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

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

B. Scholkopf, D. Janzing, J. Peters, E. Sgouritsa, K. Zhang, and J. Mooij. 2012. On causal and anticausal learning. In *International Conference on Machine Learning (ICML)*.

Rico Sennrich, B. Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. *ArXiv*, abs/1511.06709.

Damien Teney, Ehsan Abbasnejad, and A. V. Hengel. 2020. Learning what makes a difference from counterfactual examples and gradient supervision. *ArXiv*, abs/2004.09034.

Lifu Tu, Garima Lalwani, Spandana Gella, and He He. 2020. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Towards debiasing NLU models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Victor Veitch, Alexander D'Amour, Steve Yadlowsky, and Jacob Eisenstein. 2021. Counterfactual invariance to spurious correlations: Why and how to pass stress tests.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. 2018a. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Y. Wang, B. Dai, L. Kong, X. Ma, S. M. Erfani, J. Bailey, S. Xia, L. Song, and H. Zha. 2018b. Learning deep hidden nonlinear dynamics from aggregate data. In *Uncertainty in Artificial Intelligence (UAI)*.

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

Jason Wei and K. Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *ArXiv*, abs/1901.11196.

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

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

Ran Zmigrod, Sabrina J. Mielke, H. Wallach, and Ryan Cotterell. 2019. Counterfactual data augmentation for mitigating gender stereotypes in languages with rich morphology. In *ACL*.

A Proof for the Linear Model

Proof for Proposition 1. In this section, we give the proof for the toy example, where we show that $\ell(\hat{w}_{\text{inc}}) > \ell(\hat{w})$ if $\|\mu_{r2}\| > \|\mu_s\|$ assuming all variables have unit variance and $\|\mu_r\| = 1$ (i.e. the model trained on incomplete edits has higher error than model trained on unaugmented data).

Given the definition in (8), we get (denote $\text{Cov}(x, x)$ by M):

$$\begin{aligned}
\ell(\hat{w}) &= \mathbb{E}_{x\sim\mathcal{D}}\left[(w_{\text{inv}}^T x - \hat{w}^T x)^2\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[(\mu_r^T x_r - \mu^T (M^{-1})^T x)^2\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}[\mu_r^T x_r x_r^T \mu_r + \mu^T (M^{-1})^T x x^T M^{-1}\mu - 2\mu_r^T x_r x^T M^{-1}\mu] \qquad (13)
\end{aligned}$$

Note that. $\mathbb{E}_{x\sim\mathcal{D}}\left[xx^T\right] = M$, $\mathbb{E}_{x\sim\mathcal{D}}\left[x_r x_r^T\right] = I$ and $\mathbb{E}_{x\sim\mathcal{D}}\left[x_r x^T\right] = [I \;\; \mu_r\mu_s^T]$. Plugging this into the previous equation and simplifying we get:

$$\ell(\hat{w}) = \|\mu_r\|^2 - [\mu_r^T, \mu_s^T(2\|\mu_r\|^2 - 1)]M^{-1}\mu \qquad (14)$$

Since M is a block matrix, we can write its inverse as :

$$M^{-1} = \begin{bmatrix} (I - \mu_r\mu_s^T\mu_s\mu_r^T)^{-1} & -(I - \mu_r\mu_s^T\mu_s\mu_r^T)^{-1}\mu_r\mu_s^T \\ -(I - \mu_s\mu_r^T\mu_r\mu_s^T)^{-1}\mu_s\mu_r^T & (I - \mu_s\mu_r^T\mu_r\mu_s^T)^{-1} \end{bmatrix} \qquad (15)$$

Note that I here refers to the identity matrix of compatible size (i.e. either d_r or d_s dimensional). Plugging this in (14) and simplifying gives us:

$$\begin{aligned}
\ell(\hat{w}) = \|\mu_r\|^2 - \mu_r^T(I - \|\mu_s\|^2\mu_r\mu_r^T)^{-1}\mu_r(1 - \|\mu_s\|^2) - \\
\mu_s^T(I - \|\mu_r\|^2\mu_s\mu_s^T)^{-1}\mu_s(2\|\mu_r\|^2 - 1)(1 - \|\mu_r\|^2)
\end{aligned}$$

Since we have assumed that $\|\mu_r\| = 1$, we get:

$$\ell(\hat{w}) = 1 - \mu_r^T(I - \|\mu_s\|^2\mu_r\mu_r^T)^{-1}\mu_r(1 - \|\mu_s\|^2) \qquad (16)$$

Now note that $(I - \|\mu_s\|^2\mu_r\mu_r^T)$ is a rank-one perturbation of the identity matrix, and hence we can use the Sherman-Morrison formula to simplify:

$$(I - \|\mu_s\|^2\mu_r\mu_r^T)^{-1} = I + \alpha\|\mu_s\|^2\mu_r\mu_r^T \qquad (17)$$

where $\alpha > 0$ is a constant. Simplifying using this further, we get:

$$\begin{aligned}
\ell(\hat{w}) &= 1 - \mu_r^T(I + \alpha\|\mu_s\|^2\mu_r\mu_r^T)\mu_r(1 - \|\mu_s\|^2) \\
&= 1 - (\|\mu_r\|^2 + \alpha\|\mu_s\|^2\|\mu_r\|^4)(1 - \|\mu_s\|^2) \\
&= 1 - (1 + \alpha\|\mu_s\|^2)(1 - \|\mu_s\|^2) \\
&= \alpha\|\mu_s\|^4 + (1 - \alpha)\|\mu_s\|^2 \qquad (18)
\end{aligned}$$

For the incomplete edits, we have $\hat{w}_{\text{inc}} = [\Sigma_{r1}^{-1}\mu_{r1}, 0]$ giving us:

$$\begin{aligned}
\ell(\hat{w}_{\text{inc}}) &= \mathbb{E}_{x\sim\mathcal{D}}\left[(w_{\text{inv}}^T x - \hat{w}_{\text{inc}}^T x)^2\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[(\mu_{r2}^T x_{r2})^2\right] \\
&= \mathbb{E}_{x\sim\mathcal{D}}\left[\mu_{r2}^T x_{r2} x_{r2}^T \mu_{r2}\right] \\
&= \|\mu_{r2}\|^2 \qquad (19)
\end{aligned}$$

Test Set	Size (NLI)	Size (QA)
`lexical`	406	314
`resemantic`	640	332
`negation`	80	268
`quantifier`	206	80
`insert`	376	118
`delete`	250	-

Table 6: Size of the tests sets corresponding to the different perturbation types for both NLI and QA. For QA, the number of examples in `delete` were extremely small and hence we do not use that perturbation type for QA.

Train Data	All types	`lexical`	`insert`	`resemantic`	`quantifier`	`negation`	`delete`
SNLI seed	$67.84_{0.84}$	$75.16_{0.32}$	$74.94_{1.05}$	$\mathbf{76.77}_{0.74}$	$\mathbf{74.36}_{0.21}$	$\mathbf{69.25}_{2.09}$	$65.76_{2.34}$
SNLI seed (subsamples)	$64.87_{1.02}$	$75.06_{1.89}$	$71.38_{2.30}$	$73.84_{1.60}$	$69.12_{3.17}$	$66.75_{2.87}$	$63.60_{2.44}$
`lexical`	$70.44_{1.07}$	$\mathbf{81.81}_{0.99}$	$74.04_{1.04}$	$74.93_{1.16}$	$72.42_{1.58}$	$68.75_{2.16}$	$67.04_{3.00}$
`insert`	$66.00_{1.41}$	$71.08_{2.53}$	$\mathbf{78.98}_{1.58}$	$71.74_{1.53}$	$68.15_{0.88}$	$57.75_{4.54}$	$68.80_{2.71}$
`resemantic`	$\mathbf{70.80}_{1.68}$	$77.23_{2.35}$	$76.59_{1.12}$	$75.40_{1.44}$	$70.77_{1.04}$	$67.25_{2.05}$	$\mathbf{70.40}_{1.54}$

Table 7: Results for the different perturbation types in NLI with multiple subsamples of the dataset. (denotes *aligned test sets*). We observe that there is variance across different subsamples, but the majority of the trends reported in Section 3.3 still hold true.

Now, if $\|\mu_s\| < \|\mu_{r2}\|$, then $\|\mu_s\| < 1$ (since $\|\mu_{r2}\| < \|\mu_r\| = 1$). Thus, $\|\mu_s\|^4 < \|\mu_s\|^2$, giving $\ell(\hat{w}) < \|\mu_s\|^2 < \|\mu_{r2}\|^2 = \ell(\hat{w}_{\text{inc}})$.

□

B Additional Experiments & Results

Here, we report more details on the experiments as well as present some additional results.

B.1 Experiment Details

For NLI, models are trained for a maximum of 10 epochs, and for QA all models are trained for a maximum of 5 epochs (convergence is faster due to the larger dataset size). The best model is selected by performance on a held-out development set, that includes examples from the same perturbation type as in the training data.

B.2 Dataset Details

The size of the training datasets and how they are constructed are described in Section 3.2. Here, we give more details on the size of the various test sets used in the experiments. The size of the CAD datasets for the different perturbation types are given Table 6 for both NLI and QA. Note that all test sets contain paired counterfactual examples, i.e. the seed examples and their perturbations belonging to that specific perturbation type.

B.3 Accounting for small dataset sizes

The experiments in Section 3.2 were run for 5 different random initializations, and we report the mean and standard deviation across the random seeds. For completeness, we also report results when using different subsamples of the SNLI dataset. Table 7 shows the mean and standard deviation across 5 different subsamples, along with the rest of the results which were presented in Section 3.3. We observe that even though there is variance in results across the different subsamples, majority of the trends reported in 3.3 are consistent across the different subsamples — CAD performs well on aligned test sets, but does not necessarily generalize to unaligned test sets.

To account for the small dataset sizes, we also ran an experiment using the NLI CAD dataset analogous to the QA setup—using a larger number of SNLI examples (7000) and replace a small percentage of them

Train Data	All types	lexical	insert	resemantic	quantifier	negation	delete
SNLI seed	$71.41_{0.40}$	$79.90_{1.00}$	$78.08_{0.49}$	$79.84_{1.17}$	$75.92_{1.17}$	$77.25_{2.42}$	$70.88_{0.68}$
lexical	$73.10_{0.56}$	$\mathbf{83.54}_{0.91}$	$77.28_{0.64}$	$80.81_{0.47}$	$75.72_{0.86}$	$\mathbf{78.00}_{1.69}$	$70.72_{1.46}$
insert	$72.91_{0.54}$	$80.39_{0.88}$	$\mathbf{78.93}_{0.66}$	$80.56_{0.76}$	$\mathbf{76.89}_{0.84}$	$77.25_{2.66}$	$71.43_{2.40}$
resemantic	$\mathbf{73.44}_{0.33}$	$81.23_{0.64}$	$77.97_{0.51}$	$\mathbf{81.06}_{0.49}$	$76.60_{1.42}$	$75.75_{2.03}$	$\mathbf{73.84}_{1.25}$

Table 8: Results for the different perturbation types in NLI with larger dataset sizes, with 10% of the data being the perturbations (denotes *aligned test sets*).

(10%) with perturbations of the corresponding perturbation type. We ensure that the original examples from which the perturbations were generated are also present in the dataset. Thus, all experiments will have much larger dataset sizes than before (7000 vs 1400), while still using counterfactual examples generated only by one specific perturbation type. The results for this experiment are reported in Table 8. We observe that CAD still performs best on aligned test sets but only marginally — this happens since a large fraction of the dataset (90%) is similar across all experiments. Although CAD performs worse on unaligned test sets than the aligned test sets, it does not necessarily perform worse than the SNLI baseline — this happens since the larger number of seed examples will implicitly regularize the model from overfitting to that specific perturbation type.