AutoCAD: Automatically Generate Counterfactuals for Mitigating Shortcut Learning

Jiaxin Wen[1,2*], **Yeshuang Zhu**[3], **Jinchao Zhang**[3], **Jie Zhou**[3], **Minlie Huang**[1,2,†]
[1]The CoAI group, Tsinghua University, Beijing, China
[2]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[3]Pattern Recognition Center, WeChat AI, Tencent Inc, China
wenjx22@mails.tsinghua.edu.cn, aihuang@tsinghua.edu.cn
{yshzhu,dayerzhang,withtomzhou}@tencent.com

Abstract

Recent studies have shown the impressive efficacy of counterfactually augmented data (CAD) for reducing NLU models' reliance on spurious features and improving their generalizability. However, current methods still heavily rely on human efforts or task-specific designs to generate counterfactuals, thereby impeding CAD's applicability to a broad range of NLU tasks. In this paper, we present AutoCAD, a fully automatic and task-agnostic CAD generation framework. AutoCAD first leverages a classifier to unsupervisedly identify rationales as spans to be intervened, which disentangles spurious and causal features. Then, AutoCAD performs controllable generation enhanced by unlikelihood training to produce diverse counterfactuals. Extensive evaluations on multiple out-of-domain and challenge benchmarks demonstrate that AutoCAD consistently and significantly boosts the out-of-distribution performance of powerful pre-trained models across different NLU tasks, which is comparable or even better than previous state-of-the-art human-in-the-loop or task-specific CAD methods. The code is publicly available at https://github.com/thu-coai/AutoCAD.

	Original
Premise	A group of men riding bicycles in a line.
Hypothesis	The men riding together.
Label	Entailment
	AutoCAD
Premise	A group of men riding bicycles in a line.
Hypothesis	The men are professionals.
Label	Neutral
Premise	A group of men riding bicycles in a line.
Hypothesis	The men ride to work.
Label	Neutral
Premise	A group of men riding bicycles in a line.
Hypothesis	The men riding horses.
Label	Contradiction
Premise	A group of men sitting in a crowded cafe.
Hypothesis	The men riding together.
Label	Contradiction
Premise	A group of men riding separately in a crowded bus.
Hypothesis	The men riding together.
Label	Contradiction
	...

Table 1: Examples of original NLI data and counterfactual data generated by AutoCAD. The identified rationales in the original data and generated spans in the counterfactual data are highlighted in colors along with the corresponding labels.

1 Introduction

State-of-the-art NLU models have achieved impressive in-distribution performance, even surpassing humans on many benchmarks such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019). However, these apparently powerful NLU models are known to suffer from shortcut learning, i.e., learning spurious features in datasets instead of actually solving the underlying tasks, thereby leading to unsatisfactory generalizability (Geirhos et al., 2020). For example, in Natural Language Inference, a negation word can be a strong indicator of contradiction, and a high rate of word overlap between the premise and the hypothesis can be a strong indicator of entailment (Gururangan et al., 2018; Naik et al., 2018). This phenomenon hinders the practical application of NLU models.

Among the recent attempts to mitigate shortcut learning in deep neural networks, Counterfactually Augmented Data (CAD) attracts increasing attention due to its simplicity and effectiveness (Kaushik et al., 2019). Specifically, CAD are created through altering the ground-truth labels of original examples by manipulating causal features. By adding CAD to the original dataset, we can reduce spurious correlations between non-causal features and labels. The shortcut learning problem is hence mitigated from a data-centric perspective, yielding

† Corresponding author
* This work was done when Jiaxin Wen was an intern at WeChat AI.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2302–2317
December 7-11, 2022

classifiers with better generalizability.

Different from the traditional label-preserving data augmentation methods, e.g., simply replacing random words with their synonyms, the creation of CAD is more challenging as it requires precisely disentangling spurious and causal features, as well as making proper changes to alter the label. To tackle this problem, many existing studies still rely on human efforts to create CAD (Kaushik et al., 2019; Gardner et al., 2020) or label the generated CAD (Wu et al., 2021), which is costly and time-consuming. Moreover, manually-curated CAD may even exacerbate existing spurious patterns or introduce new ones due to the lack of diversity in annotators' edits (Huang et al., 2020; Joshi and He, 2021). While task-specific methods can be designed to automatically generate counterfactuals for sentiment analysis (Yang et al., 2021) and open-domain QA(Paranjape et al., 2021), the design and effectiveness of a fully automatic and task-agnostic CAD generator are still under-explored.

In this paper, we present AutoCAD, a fully automatic and task-agnostic CAD generation framework, which only takes the original NLU dataset as input and can generate diverse counterfactuals guided by any given target label. Our framework design completely eliminates the need for human efforts, task-specific designs, or parallel counterfactually augmented data for supervised training. Specifically, AutoCAD first leverages the gradients of a trained classifier to automatically identify rationales as spans to be intervened. Then, AutoCAD formulates counterfactual generation as a label-controlled text-infilling task with the help of a large-scale sequence-to-sequence language model. AutoCAD further introduces unlikelihood training (Welleck et al., 2019) to improve the controllability of counterfactual generation.

We study the effectiveness of AutoCAD on two widely-adopted NLU tasks: Natural Language Inference and Sentiment Analysis. We extend the original training data with counterfactuals automatically generated by AutoCAD and evaluate the generalizability of state-of-the-art NLU models on multiple out-of-domain and challenge benchmarks. Extensive experiments demonstrate that AutoCAD consistently and significantly improves the out-of-distribution performance across different tasks and different powerful pre-trained models. AutoCAD outperforms previous non-CAD data augmentation baselines by a large margin, especially on the challenging benchmarks where shortcut learning behavior is amplified. It also achieves comparable or even better results than previous state-of-the-art human-in-the-loop or task-specific CAD methods.

2 Related Work

2.1 Mitigating Shortcut Learning of NLU Models

There are two lines of research towards mitigating shortcut learning of NLU models: model-centric and data-centric. Model-centric methods focus on reducing the reliance on spurious features during the training phase of NLU models. Clark et al. (2019); He et al. (2019); Mahabadi et al. (2019) propose to ensemble a bias-only model with the main model based on the product-of-experts(PoE) framework to suppress it from focusing on the known dataset-specific bias. Utama et al. (2020); Du et al. (2021) further propose general methods to quantify the shortcut degree of each sample without the prior knowledge of bias and then debias NLU models through re-weighting or confidence regularization during training. Instead of incorporating additional modules or training objectives, data-centric methods focus on intrinsically reducing the spurious features in datasets. Wu et al. (2022) proposes a filtering mechanism based on z-statistics (Gardner et al., 2021) for removing data samples that contribute to spurious features. And CAD also falls into this category.

2.2 Counterfactually Augmented Data

Kaushik et al. (2019); Gardner et al. (2020) employ human annotators to create CAD by manually rewriting the original datasets. However, manual rewrites are not only time-consuming and expensive but also may exacerbate existing spurious features or introduce new ones due to the lack of diversity in annotators' edits (Huang et al., 2020; Joshi and He, 2021). To alleviate this issue, Wu et al. (2021) propose POLYJUICE, a task-agnostic GPT2-based counterfactual generator, which is fine-tuned on parallel original-counterfactual pairs collected from multiple datasets to allow for control codes such as *negation*, *insert* and *delete*. However, since POLYJUICE is an untargeted counterfactual generator, human annotators are still needed to label the generated counterfactuals. There are also some task-specific methods to automatically generate counterfactuals for sentiment analysis (Yang et al.,

2021) and open-domain QA(Paranjape et al., 2021). Similar to our work, Madaan et al. (2021); Ross et al. (2020) also aims to design a task-agnostic automatic counterfactual generator. Our work is distinguished from these mainly in that we not only evaluate the performance of the proposed generator in controllability and diversity but also conduct extensive experiments on multiple out-of-domain and challenge benchmarks to thoroughly investigate the efficacy of the automatically generated CAD for improving the generalizability of powerful pre-trained NLU models across different tasks.

2.3 Controllable Text Generation

Controllable text generation aims to generate texts aligning with the desired attribute and hence is an essential component of automatic CAD generation. The most straightforward and commonly used approach is to directly fine-tune a generative model with the concatenation of the text and the targeted attribute, which is also known as Class-Conditional Language Model (CCLM) (Keskar et al., 2019). Another line of work achieves attribute control during the decoding process without updating parameters of large-scale language models, which reduces computation costs (Dathathri et al., 2019; Krause et al., 2020; Liu et al., 2021; Yang and Klein, 2021). In general, CCLM performs better in text quality and inference speed but worse in controllability. We thus base our generator on CCLM and further introduce unlikelihood training to enhance its controllability.

3 Methodology

3.1 Task Definition

Let $\{(x_i, y_i)\}$ be the training set of a classification task, where x_i can either be a text or a text pair, and $y_i \in \mathcal{Y}$ is the corresponding label.

For each (x_i, y_i) and a given target label $\hat{y}_i \neq y_i$, we define the counterfactual generation task as to generate one or more counterfactual example $\hat{x}_i$ that meet the following requirements. 1) ***Label flipping***: $\hat{x}_i$ should be in accord with the target label $\hat{y}_i$. 2) ***Fluent***: $\hat{x}_i$ should be coherent and grammatically correct. 3) ***Minimal change***:

Ideally, $\hat{x}_i$ should only intervene in the rationales (causal features) to disentangle them from the spurious features. The requirement of *minimal change* is positively correlated with *label flipping* since it is less likely to flip the original label y_i without touching its rationales. 4) ***Diverse***: the changes from x_i to $\hat{x}_i$ should be diverse among the dataset, especially when giving the same target label $\hat{y}_i$. Otherwise, the generated counterfactuals may exacerbate existing spurious features or introduce new spurious features (Huang et al., 2020; Joshi and He, 2021).

To tackle the problem, AutoCAD first leverages a trained classifier to unsupervisedly identify rationales as spans to be intervened, eliminating the need for human efforts or task-specific designs. Then, AutoCAD formulates counterfactual generation as a label-controlled text-infilling task and introduces unlikelihood training (Welleck et al., 2019) to improve the controllability of counterfactual generation, eliminating the need for parallel CAD for supervised training. After generating $\hat{x}_i$, AutoCAD further uses the classifier to post-select the generated samples. Figure 1 shows the framework overview of AutoCAD.

3.2 Identifying Rationales

To meet the requirement of *minimal change* and *label flipping*, we need to select the spans to be intervened carefully. Ideally, changing the words exactly belonging to rationales will be the most effective approach. However, the golden rationales are unavailable without human efforts. Therefore, we adopt the existing task-agnostic post-hoc explanation methods to automatically identify rationales. There are two mainstream categories of post-hoc explanation methods: perturbation-based (Ribeiro et al., 2016; Li et al., 2016) and gradient-based (Simonyan et al., 2013; Li et al., 2015a). In this work, we implement AutoCAD with the gradient norm (Li et al., 2015a), and our framework can be easily adapted to any other rationale extraction method.

Formally, given a classification model trained on the original dataset and a data sample (x, y), where $x = (w_1, w_2, \cdots, w_m) = (t_1, t_2, \cdots, t_n)$ is the text input with m words and n tokens, and y is the classification label, we first calculate the gradient of the model output with respect to the embedding e_i of each token t_i in the input x:

$$g(t_i) = \nabla_{e_i} P(y|x)$$

where $P(y|x)$ is the output logit for label y given the input x.

Then we obtain the saliency score s_{t_i} of each token t_i by taking the $l2$ norm of the gradient and re-normalizing it along all the input tokens:

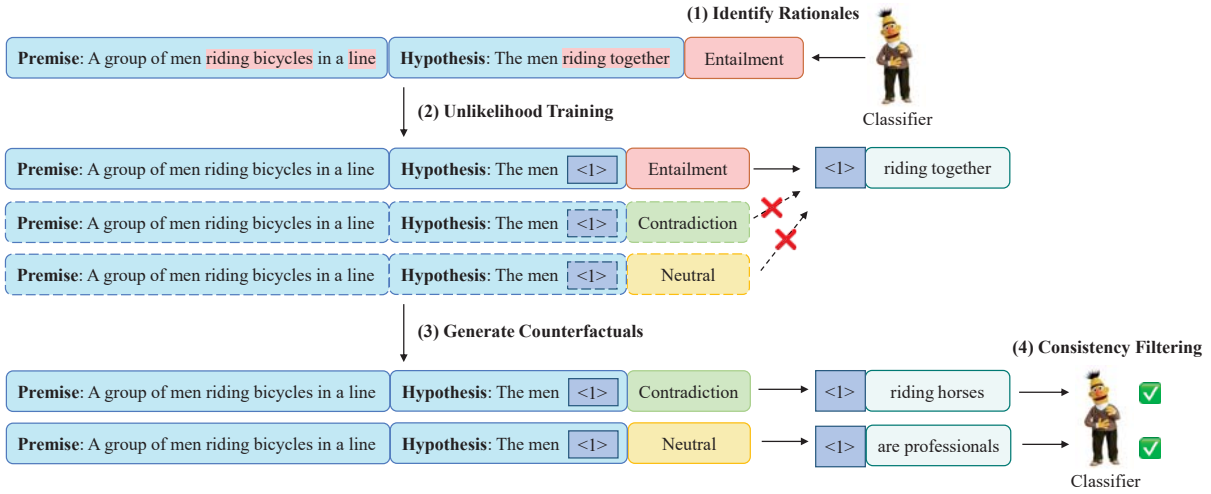

Figure 1: Overview of AutoCAD. The framework consists of four steps: (1) leverage a classifier to unsupervisedly identify rationales as spans to be intervened (masked); (2) train a controllable text-infilling generator enhanced by unlikelihood training; (3) generate counterfactuals according to a flipped target label; (4) post-select the generated samples based on the agreement of the target label and the predicted label of the classifier.

$$s_{t_i} = \frac{||g(t_i)||_2}{\sum_{j=1}^{n} ||g(t_j)||_2}$$

Considering that some tokenization algorithms adopted in widely used NLU Models, e.g., byte-level Byte-Pair-Encoding (BPE) in RoBERTa (Liu et al., 2019), would split a single word w_i into K sub-tokens $(t_i^1, t_i^2, \cdots, t_i^K)$, we further derive the word-level saliency score s_{w_i} as the largest saliency score of its sub-tokens:

$$s_{w_i} = \max_{k=1}^{K} s_{t_i^k}$$

After obtaining the word-level saliency score, we select the top $\pi\%$ words with the largest s_{w_i} values as rationales, where π is a threshold hyperparameter. Note that the classifier we used is trained on the original dataset and thereby is also exposed to spurious features. In order to mitigate error propagation, we only select those samples for which the model predictions are correct. We assume the model is more likely to exploit the golden rationales in these samples than the rest.

3.3 Unlikelihood Training for Label-Controlled Text Infilling

In this section, we aim to train a generator that can modify the identified rationales in a diverse way and produce counterfactual data in accord with the new label. As we do not have parallel CAD for supervised training like POLYJUICE, we formulate counterfactual generation as a label-controlled text-infilling task, i.e., to generate more variations for the masked rationale spans in accord with a given flipped label. Moreover, this task formulation also ensures that all the identified non-rationales will remain unchanged in the generated counterfactuals.

We base our generation model on an encoder-decoder model - T5 (Raffel et al., 2019) particularly - for two reasons. 1) Controllability and diversity are both crucial for the effectiveness of CAD. As revealed by Kumar et al. (2020), texts generated by encoder-decoder models achieve a good balance between controllability and diversity compared with encoder-only models like BERT (Devlin et al., 2018) or decoder-only models like GPT (Radford et al.). 2) T5 is pre-trained with the same text-infilling objective, which mitigates the gap between pre-training and fine-tuning. Thus we can take better advantage of knowledge in pre-trained language models for generating counterfactuals.

In order to realize label-controlled generation, the common training objective of CCLM is to minimize the negative log-likelihood loss $\mathcal{L}_{MLE}$ of reconstructing the text output conditioned on the given label. In AutoCAD, the loss is as follows:

$$\mathcal{L}_{MLE} = -\sum_{t=1}^{|z|} \log P(z_t | z_{<t}, x - z, y)$$

where z denotes the identified rationales, x is the original text input, $x - z$ is the rationale-corrupted text input, and y is the classification label.

However, using only the likelihood training objective may lead the model to over-focus on the text input $x - z$ and ignore the control label y.

To alleviate this issue, we introduce unlikelihood training (Welleck et al., 2019) to suppress the model from always generating the original rationales when a perturbed label $\hat{y}$ is given. Formally, the unlikelihood loss $\mathcal{L}_{UL}$ is computed as follows:

$$\mathcal{L}_{UL} = -\sum_{\hat{y}\in\mathcal{Y},\hat{y}\neq y}\sum_{t=1}^{|z|}\log(1-P(z_t|z_{<t}, x-z, \hat{y}))$$

Then we derive the overall loss function as follows:

$$\mathcal{L} = \mathcal{L}_{MLE} + \alpha * \mathcal{L}_{UL}$$

where the coefficient α is a hyperparameter to control the strength of the unlikelihood loss.

Intuitively, minimizing the unlikelihood loss $\mathcal{L}_{UL}$ improves the model's sensitivity to the control label and thus results in better controllability on generation. However, we note that the token z_t to be penalized needs to be carefully selected. As non-rationales have no strong association with any control label y, penalizing them may accidentally introduce noise to the generator and degrade its performance, which is validated in our experiment results in Section 4.4.

3.4 Consistency Filtering

Since we aim at a task-agnostic and fully-automatic counterfactual generation framework, the controllability of the generator could be insufficient when applied to some challenging NLU tasks such as Natural Language Inference. Therefore, we further leverage the classification model used in Section 3.2 to post-select the generated output. Specifically, we deem a generated counterfactual as valid if its target label is consistent with the prediction of the classifier. After consistency filtering, we finally obtain the counterfactually augmented data.

4 Experiments

4.1 Benchmark Tasks

We evaluate our method on two widely-adopted NLU tasks, Natural Language Inference (NLI) and Sentiment Analysis (SA). We select the SST-2 dataset which contains 8,544 examples (Socher et al., 2013) and a subset from the SNLI dataset which contains 20,000 examples (Bowman et al., 2015) as the original in-domain dataset. And since we focus on measuring the generalizability of the model after data augmentation, we further conduct extensive evaluations on multiple out-of-domain and challenge benchmarks for each task. For more details about the evaluation benchmarks, please refer to Appendix A.2.

4.2 Baselines

We compare AutoCAD with the following baselines. For all the automatic data augmentation methods, we augment each sample once unless otherwise specified.

Synonym Replacement A proportion of r words are chosen from the sentence (excluding the stopwords), and each chosen word is independently replaced with its synonyms based on WordNet (Zhang et al., 2015). The r is set to 30% in our experiment.

Back Translation The sentence is first translated to another language, and then back-translated to the source language (Sennrich et al., 2015).

BERT-MLM A proportion of r words are chosen from the sentence excluding the stopwords, and each chosen word is independently replaced with the top k words predicted by a pre-trained BERT model (Jiao et al., 2019). The r is set to 30% and k is set to 100 in our experiment.

Sentiment-CAD Sentiment-CAD (Yang et al., 2021) is the previous state-of-the-art method to automatically generate counterfactuals for sentiment analysis. Sentiment-CAD uses an external sentiment word dictionary and a sentiment classifier to identify rationales. For each word in the identified rationales, it either removes the word (usually a negation word such as "not" and "no") or generates an alternative word with an opposing sentiment polarity according to the sentiment word dictionary using BERT-MLM. However, there are two drawbacks of this method. First, the reliance on the external word dictionary makes Sentiment-CAD a task-specific method that may face the challenge when handling complex data without explicit sentiment words. Second, the diversity of generated data is limited as the encoder-only model used in Sentiment-CAD can only generate fix-length alternatives that are further constrained by the predefined dictionary.

Human-CAD The Human-CAD dataset (Kaushik et al., 2019) is created by employing human annotators to rewrite a subset of the SNLI dataset. In our experiment, we ensure that our training dataset includes all the samples in the subset.

We also compare AutoCAD with other methods that are not specially designed for CAD (see Appendix A.4).

4.3 Implementation Details

We adopt the state-of-the-art pre-trained NLU model $\text{RoBERTa}_{\text{LARGE}}$ (Liu et al., 2019) as the classifier to identify rationales and do consistency filtering, and T5_{LARGE} (Raffel et al., 2019) as the generator. For training the classifier, we set the batch size to 32, the initial learning rate of the AdamW optimizer to 1e-5, and the maximum training epoch to 20. We select the best checkpoint based on the accuracy of the in-domain validation set. For training the generator, we set the batch size to 8, the initial learning rate of AdamW to 1e-5, and the maximum training epoch to 10 with an early stopping mechanism. We set $\alpha = 1$ for Natural Language Inference, and $\alpha = 0.3$ for Sentiment Analysis. We select the best checkpoint based on the perplexity of the validation set. We generate counterfactuals using nucleus sampling (Holtzman et al., 2019) with $p = 0.9$ and temperature $= 0.7$.

4.4 Main Results

The results on NLI and SA are shown in Table 2. It can be seen that methods based on CAD generally outperform non-CAD methods. In particular, on NLI, which is a more challenging NLU task, we observe that all the non-CAD baselines result in negative in-distribution and out-of-distribution performance, except in only a few cases where a slight improvement can be achieved. In contrast, AutoCAD consistently and significantly improves out-of-distribution performance while maintaining or slightly improving the in-distribution performance across different tasks and different pre-trained models. Especially that AutoCAD achieves impressive results on the challenge sets where shortcut learning behavior is amplified, demonstrating its effectiveness for mitigating shortcut learning. Furthermore, it also achieves comparable or even better performance than Human-CAD and Sentiment-CAD, while eliminating the need for any human effort or task-specific design.

4.5 Validity of the Identified Rationales

We first measure the alignment of model-identified rationales with human rationales from e-SNLI (Camburu et al., 2018; DeYoung et al., 2019). The token-level macro-F1 score is 0.46. Following Kaushik et al. (2020), we further conduct experiments to verify the validity of the identified rationales. We randomly mask $r\%$ tokens in the identified rationales and non-rationales respectively, and observe performance changes of the NLU model trained on these two differently noised datasets. If the identified rationales can represent causal features, masking these rationales is expected to result in worse model performance than masking non-rationales. Furthermore, the difference should get amplified as the noising ratio $r\%$ increases. We conduct experiments on NLI using $\text{BERT}_{\text{BASE}}$. As suggested by Kaushik et al. (2020), we select only those premise-hypothesis pairs altogether with more than 9 tokens marked as rationales, eliminating the length imbalance between rationales and non-rationales. As shown in Figure 2, we observe a significantly sharper decrease in accuracy when adding noise to the identified rationales in all the test datasets, covering in-domain, out-of-domain, and challenge settings. The results demonstrate the internal validity of the identified rationales.

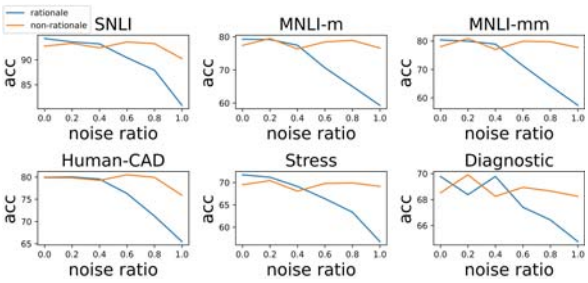

Figure 2: Changes in accuracy as we add noise to the unsupervisedly identified rationales or non-rationales.

4.6 Ablation Study

To further investigate the influence of each component in AutoCAD, we run an ablation study on NLI and report the metrics of generation quality, including controllability measured by $label\ flipping\ rate$, and diversity by $distinct\text{-}n$ (Madaan et al., 2021). From the results in Table 3, it can be seen that combining rationale-masking strategy and unlikelihood training achieves the best performance on controllability and diversity. Moreover, we draw the following conclusions. 1) Masking rationales instead of random spans can effec-

Method	Natural Language Inference								Sentiment Analysis					
	In-Domain	Out-of-Domain		Challenge				Avg.	In-Domain	Out-of-Domain		Challenge		Avg.
	SNLI	MNLI-m	MNLI-mm	Human-CAD	Diagnostic	Stress	Break		SST-2	IMDb	Yelp	Human-CAD	Contrast	
BERT$_{BASE}$														
Original	84.84	63.02	63.84	61.25	50.27	54.55	69.32	60.38	88.42	86.68	88.95	87.50	82.58	86.43
Synonym Rep.	84.77	64.06	64.61	61.44	49.91	57.26	68.40	60.95	87.42	86.07	90.82	87.90	83.40	87.05
Back Trans.	84.86	63.89	64.04	61.25	49.73	56.91	66.35	60.36	87.60	**89.14**	89.27	87.50	83.81	87.43
BERT-MLM	83.92	62.78	63.88	57.18	49.73	56.30	66.25	59.35	87.06	84.84	88.53	80.12	77.25	82.69
Human-CAD	85.75	66.26	66.14	70.87	51.72	57.74	79.38	65.35	-	-	-	-	-	
Sentiment-CAD	-	-	-	-	-	-	-	-	87.73	88.93	89.73	90.16	87.09	88.98
AutoCAD	**87.25**	**69.67**	**70.27**	**71.43**	**54.26**	**59.13**	**89.59**	**68.38**	88.19	88.52	**90.94**	**91.80**	**88.73**	**90.00**
BERT$_{LARGE}$														
Original	86.15	69.32	70.15	66.75	53.80	60.92	83.58	67.42	87.38	85.45	87.88	88.52	83.40	86.31
Synonym Rep.	86.77	70.85	71.70	66.87	54.53	64.27	80.69	68.15	88.00	**87.50**	88.57	**92.01**	84.43	88.13
Back Trans.	86.40	68.61	68.71	64.75	54.53	60.31	76.41	65.55	87.38	80.53	83.38	80.12	74.59	79.66
BERT-MLM	84.38	65.61	66.74	58.50	50.72	58.69	69.18	61.57	88.05	87.30	88.54	84.84	79.51	85.05
Human-CAD	86.88	70.46	69.36	**73.87**	53.89	61.78	90.37	69.96	-	-	-	-	-	
Sentimen-CAD	-	-	-	-	-	-	-	-	88.87	85.04	88.56	88.73	85.66	87.00
AutoCAD	**87.98**	**74.50**	**75.05**	73.75	**55.98**	**65.04**	**90.53**	**72.22**	87.78	86.27	**89.94**	90.98	**86.68**	**89.04**
RoBERTa$_{BASE}$														
Original	88.02	75.07	76.07	68.31	55.07	**67.09**	91.22	72.14	91.09	85.66	91.16	85.45	81.56	85.96
Synonym Rep.	87.61	73.91	75.23	67.44	55.34	65.43	84.79	70.32	89.68	87.70	93.26	90.16	87.70	89.71
Back Trans.	87.86	74.11	74.75	67.12	54.34	65.02	80.10	69.28	91.18	87.50	92.22	88.52	84.84	88.27
BERT-MLM	87.13	73.25	74.08	66.87	54.53	64.83	89.59	70.53	89.55	87.09	91.46	85.86	80.12	86.13
Human-CAD	87.42	75.85	76.01	**75.56**	56.52	64.86	90.96	73.29	-	-	-	-	-	
Sentiment-CAD	-	-	-	-	-	-	-	-	89.95	**89.95**	**93.41**	**94.26**	**91.19**	**92.21**
AutoCAD	88.11	**76.32**	**77.09**	74.56	**56.88**	65.17	**91.77**	**73.63**	90.81	88.52	92.47	92.62	88.52	90.53
RoBERTa$_{LARGE}$														
Original	89.42	80.13	80.29	73.56	58.24	67.82	90.36	75.07	91.58	88.73	94.84	88.93	87.70	90.05
Synonym Rep.	89.58	80.83	81.77	72.62	56.70	72.33	91.42	75.95	90.41	89.55	94.57	89.55	90.37	91.01
Back Trans.	88.92	78.75	78.95	72.37	59.51	67.38	90.54	74.58	89.68	91.60	95.49	92.62	91.39	92.78
BERT-MLM	89.02	76.94	77.29	67.12	55.71	68.91	86.81	72.13	90.86	92.21	**96.16**	88.31	86.48	90.79
Human-CAD	**90.23**	81.90	**82.07**	**78.50**	60.14	72.06	**93.97**	78.11	-	-	-	-	-	
Sentiment-CAD	-	-	-	-	-	-	-	-	89.82	90.34	93.60	94.67	91.80	92.35
AutoCAD	89.63	**82.25**	81.89	76.25	**61.32**	**74.14**	93.93	**78.30**	90.50	**92.83**	95.60	**94.48**	**93.03**	**93.99**

Table 2: Model Accuracy on different NLU tasks with different data augmentation methods. The best results are highlighted in **bold**. The average scores take into account the scores on all the out-of-domain and challenge test sets.

tively improve the label flipping rate, in both the training and generation phases and in different loss settings. 2) While the controllability of the generator improves after fine-tuning with standard MLE objective, combining with unlikelihood training can further boost the label flipping rate from 42.17% to 68.56%. 3) Unlikelihood training significantly improves the diversity of generation, as the generator is forced to generate under the guidance of the target label other than just generating the words seen in the original example. 4) Masking rationales can fully exploit the benefits of unlikelihood training, resulting in a substantial controllability improvement from 47.87% to 68.56%. In fact, when using random mask in unlikelihood training, we observe that the unlikelihood loss $\mathcal{L}_{UL}$ will gradually conflict with the likelihood loss $\mathcal{L}_{MLE}$.

4.7 Analysis of α in Unlikelihood Training

We investigate the effect of the coefficient α on the generator's performance on the NLI task. As shown in Figure 3, as α increases, there is a significant improvement in label flipping rate and diversity, with a modest increase in perplexity. This trend slows down after the coefficient α exceeds 1.0.

Variants (train loss)	Mask$_{train}$	Mask$_{gen}$	FR	Distinct-3/4
AutoCAD$_{notrain}$	N/A	random	23.65	0.25/0.94
AutoCAD$_{MLE}$	random	random	32.17	0.33/1.18
AutoCAD$_{MLE}$	random	rationales	34.09	0.23/0.83
AutoCAD$_{MLE}$	rationales	rationales	42.17	0.27/0.94
AutoCAD$_{MLE+UL}$	random	rationales	47.87	0.39/**1.61**
AutoCAD$_{MLE+UL}$	rationales	rationales	**68.56**	**0.40**/1.48

Table 3: Ablation study on the effect of rationales and unlikelihood training on generator's performance, conducted on SNLI. FR means label flipping rate. (Refer to Appendix A.1 for more details about the metrics.)

4.8 Comparing AutoCAD with FlipDA

We note that AutoCAD$_{notrain}$ is similar to FlipDA (Zhou et al., 2021), a concurrent work that focuses on improving in-domain performance in the low-resource scenario only and gets substantial improvement on various NLU tasks. While both AutoCAD$_{notrain}$ and FlipDA randomly select spans to be intervened and leverage the vanilla T5 model to fill in the blank, FlipDA has a more specific design on prompt engineering and post-

Method(Augmenting Times)	**In-Domain**	**Out-of-Domain**		**Challenge**				**Avg. (%)**
	SNLI	MNLI-m	MNLI-mm	CAD	Diagnostic	Stress	Break	
Original	84.84	63.02	63.84	61.25	50.27	54.55	69.32	60.38
FlipDA (10)	86.07	68.81	69.27	67.19	53.17	57.73	82.49	66.44
AutoCAD (1)	87.25	69.67	70.27	71.43	54.26	59.13	89.59	68.38
AutoCAD (10)	**87.61**	**72.35**	**72.96**	**72.31**	**55.71**	**60.14**	**92.19**	**70.94**

Table 4: Comparison between AutoCAD and FlipDA, conducted on $\text{BERT}_{\text{BASE}}$. (1) and (10) after each method mean the number of augmenting times. CAD generated by AutoCAD are more sample efficient than those by FlipDA.

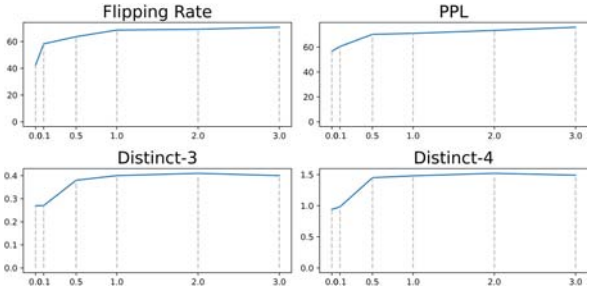

Figure 3: The effect of the coefficient α on the performance of the generator trained on SNLI.

selection strategy. Therefore, we adopt the same setting from their paper to fully exploit its performance: (1) we use "*{premise} ? Yes/No/Maybe . {hypothesis}*" as the prompt. (2) we augment $N = 10$ times for each example in the original dataset. (3) we select the generated counterfactual with the largest probabilities for the target label.

As shown in Table 4, AutoCAD consistently and substantially outperforms FlipDA while using fewer counterfactuals after augmenting only once. Moreover, augmenting $N = 10$ times with AutoCAD will further boost the performance of NLU models. The results indicate that the counterfactuals generated by AutoCAD are more informative and sample efficient compared with FlipDA. We conjecture that the random mask strategy and the weaker controllability of FlipDA may lead to more label-preserving adversarial examples, and thereby introduce more label noise.

4.9 Case Study

We present two cases from our NLU benchmark tasks in Figure 4. We can observe that the NLU model trained on the original data over-relies on words like "not" and "refuses", leading to wrong predictions. In contrast, counterfactually augmented data generated by AutoCAD effectively mitigates this phenomenon and successfully corrects the model predictions. We also present multiple generation examples in Appendix A.6 to demonstrate that AutoCAD can generate diverse counterfactuals across different tasks.

4.10 Discussion

By disentangling instance-level causal and non-causal features, CAD could implicitly mitigate unknown statistical spurious features in dataset-level (Kaushik et al., 2019, 2020). The assumption here is different from that the prior of spurious features should be known and characterized with researchers' task-specific insights (Clark et al., 2019; He et al., 2019; Mahabadi et al., 2019). The latter has several limitations: 1) Datasets are likely to contain unknown spurious features which are hard to define. 2) The spurious features are dataset-specific.

5 Conclusion

We propose AutoCAD, a fully automatic and task-agnostic counterfactually augmented data generation framework. AutoCAD combines effective rationale extraction methods and a controllable generative model enhanced by unlikelihood training, which can generate diverse counterfactuals. Extensive experiments on multiple out-of-domain and challenge benchmarks demonstrate that AutoCAD consistently and significantly improves the out-of-distribution performance of powerful pre-trained NLU models. More importantly, AutoCAD achieves comparable or even better performance than previous state-of-the-art human-in-the-loop or task-specific methods. We believe this work has broad interests in various NLU tasks.

6 Limitations

Despite the effectiveness of the gradient norm for rationale extraction in AutoCAD, we have not further explored more advanced methods such as LIME (Ribeiro et al., 2016), SHAP (Lundberg and Lee, 2017), and L2E (Situ et al., 2021). Another limitation of our work is that AutoCAD is still a

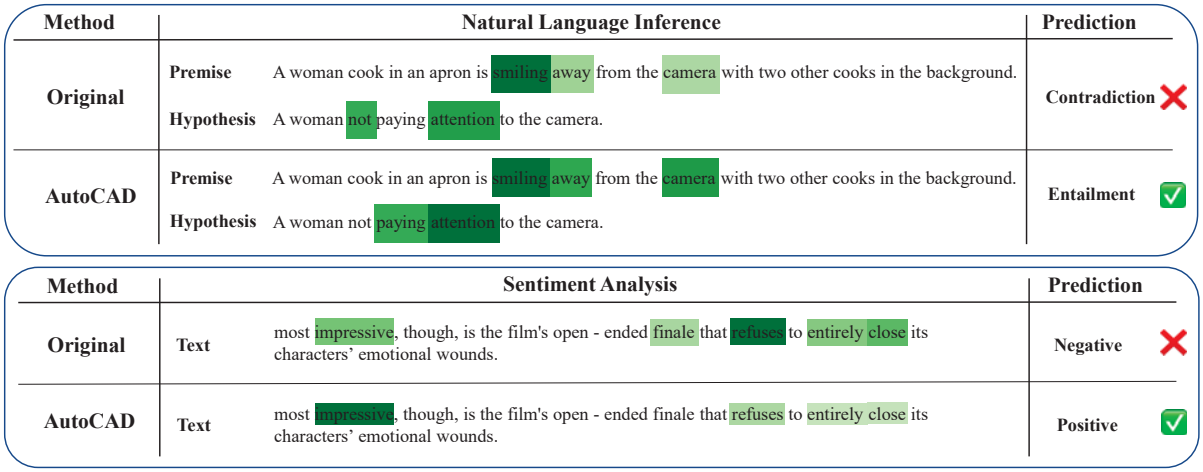

Figure 4: Examples of the effect of AutoCAD on downstream NLU tasks. We observe the prediction changes of a $\text{BERT}_{\text{BASE}}$ model trained with or without data augmentation by AutoCAD. The highlighted words are the top-5 rationales identified by the method in Section 3.2, indicating the importance of each word to the model decisions.

pipeline framework. Some recent studies attempt to jointly optimize a rationale extractor and a classifier in an end-to-end fashion (Lei et al., 2016; Chang et al., 2020; Paranjape et al., 2020; Yu et al., 2021). We will extend AutoCAD to an end-to-end framework by jointly training a rationale extractor and a counterfactual generator.

Acknowledgement

This work was supported by the National Science Foundation for Distinguished Young Scholars (with No. 62125604) and the NSFC projects (Key project with No. 61936010 and regular project with No. 61876096). This work was also supported by the Guoqiang Institute of Tsinghua University, with Grant No. 2019GQG1 and 2020GQG0005, and sponsored by Tsinghua-Toyota Joint Research Fund. We would also like to thank the anonymous reviewers for their invaluable suggestions and feedback.

References

Nabiha Asghar. 2016. Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31.

Chun-Hao Chang, George Alexandru Adam, and Anna Goldenberg. 2021. Towards robust classification model by counterfactual and invariant data generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15212–15221.

Shiyu Chang, Yang Zhang, Mo Yu, and Tommi Jaakkola. 2020. Invariant rationalization. In *International Conference on Machine Learning*, pages 1448–1458. PMLR.

Seungtaek Choi, Myeongho Jeong, Hojae Han, and Seung won Hwang. 2022. C2l: Causally contrastive learning for robust text classification. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):10526–10534.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: ensemble based methods for avoiding known dataset biases. *arXiv preprint arXiv:1909.03683*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C Wallace. 2019. Eraser: A benchmark to evaluate rationalized nlp models. *arXiv preprint arXiv:1911.03429*.

Yana Dranker, He He, and Yonatan Belinkov. 2021. Irm—when it works and when it doesn't: A test case

of natural language inference. *Advances in Neural Information Processing Systems*, 34:18212–18224.

Mengnan Du, Varun Manjunatha, Rajiv Jain, Ruchi Deshpande, Franck Dernoncourt, Jiuxiang Gu, Tong Sun, and Xia Hu. 2021. Towards interpreting and mitigating shortcut learning behavior of nlu models. *arXiv preprint arXiv:2103.06922*.

Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, et al. 2020. Evaluating models' local decision boundaries via contrast sets. *arXiv preprint arXiv:2004.02709*.

Matt Gardner, William Merrill, Jesse Dodge, Matthew E Peters, Alexis Ross, Sameer Singh, and Noah Smith. 2021. Competency problems: On finding and removing artifacts in language data. *arXiv preprint arXiv:2104.08646*.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. *arXiv preprint arXiv:1805.02266*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. *arXiv preprint arXiv:1908.10763*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

William Huang, Haokun Liu, and Samuel R Bowman. 2020. Counterfactually-augmented snli training data does not yield better generalization than unaugmented data. *arXiv preprint arXiv:2010.04762*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Nitish Joshi and He He. 2021. An investigation of the (in) effectiveness of counterfactually augmented data. *arXiv preprint arXiv:2107.00753*.

Divyansh Kaushik, Eduard Hovy, and Zachary C Lipton. 2019. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434*.

Divyansh Kaushik, Amrith Setlur, Eduard Hovy, and Zachary C Lipton. 2020. Explaining the efficacy of counterfactually augmented data. *arXiv preprint arXiv:2010.02114*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2020. Gedi: Generative discriminator guided sequence generation. *arXiv preprint arXiv:2009.06367*.

Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. Data augmentation using pre-trained transformer models.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2015a. Visualizing and understanding neural models in nlp. *arXiv preprint arXiv:1506.01066*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015b. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A Smith, and Yejin Choi. 2021. Dexperts: Decoding-time controlled text generation with experts and anti-experts. *arXiv preprint arXiv:2105.03023*.

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*.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

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.

Nishtha Madaan, Inkit Padhi, Naveen Panwar, and Diptikalyan Saha. 2021. Generate your counterfactuals: Towards controlled counterfactual generation for text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13516–13524.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2019. End-to-end bias mitigation by modelling biases in corpora. *arXiv preprint arXiv:1909.06321*.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. *arXiv preprint arXiv:1806.00692*.

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. *arXiv preprint arXiv:2005.00652*.

Bhargavi Paranjape, Matthew Lamm, and Ian Tenney. 2021. Retrieval-guided counterfactual generation for qa. *arXiv preprint arXiv:2110.07596*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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

Alexis Ross, Ana Marasović, and Matthew E Peters. 2020. Explaining nlp models via minimal contrastive editing (mice). *arXiv preprint arXiv:2012.13985*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. *Advances in neural information processing systems*, 30.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Xuelin Situ, Ingrid Zukerman, Cecile Paris, Sameen Maruf, and Gholamreza Haffari. 2021. Learning to explain: Generating stable explanations fast. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5340–5355.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Towards debiasing nlu models from unknown biases. *arXiv preprint arXiv:2009.12303*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

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*.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2019. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pages 38–45.

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*.

Yuxiang Wu, Matt Gardner, Pontus Stenetorp, and Pradeep Dasigi. 2022. Generating data to mitigate spurious correlations in natural language inference datasets. *arXiv preprint arXiv:2203.12942*.

Kevin Yang and Dan Klein. 2021. Fudge: Controlled text generation with future discriminators. *arXiv preprint arXiv:2104.05218*.

Linyi Yang, Jiazheng Li, Pádraig Cunningham, Yue Zhang, Barry Smyth, and Ruihai Dong. 2021. Exploring the efficacy of automatically generated counterfactuals for sentiment analysis. *arXiv preprint arXiv:2106.15231*.

Mo Yu, Yang Zhang, Shiyu Chang, and Tommi Jaakkola. 2021. Understanding interlocking dynamics of cooperative rationalization. *Advances in Neural Information Processing Systems*, 34:12822–12835.

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

Jing Zhou, Yanan Zheng, Jie Tang, Jian Li, and Zhilin Yang. 2021. Flipda: Effective and robust data augmentation for few-shot learning.

A Appendix

A.1 Metrics

Distinct-N It is computed as the number of distinct n-grams divided by the number of all n-grams in generated data (Li et al., 2015b).

Label Flipping Rate (FR) Ideally, it is computed as the proportion of the generated counterfactual data whose target label is consistent with its golden label. Since the golden label is unavailable without human efforts, we use the prediction of the classifier instead. Formally,

$$\text{FR} = \frac{\sum_{j=1}^{N} \mathbb{1}[\hat{y} = \arg\max_{y \in \mathcal{Y}} P(y|\hat{x})]}{N}$$

where N is the size of all the generated counterfactuals and $\mathbb{1}[\cdot]$ is the indicator function.

A.2 Details of Evaluation Benchmarks

A.2.1 Natural Language Inference

- **MNLI-m and MNLI-mm** (Williams et al., 2017): The matched and the mismatched test set of the Multi-Genre NLI dataset (MultiNLI) differ in text domains. MultiNLI is derived from ten different text genres of written and spoken English, and is more challenging compared with the Standford NLI dataset (SNLI) (Bowman et al., 2015) which is derived from only one domain.

- **Human-CAD** (Kaushik et al., 2019): The Human-CAD dataset for NLI is a manually-curated counterfactually augmented dataset created by employing human annotators to rewrite a subset of the SNLI dataset.

- **Diagnostic** (Wang et al., 2018): The Diagnostic dataset is a manually-curated test set for evaluating the model's ability on several important linguistic phenomena, such as lexical semantics and logic.

- **Stress** (Naik et al., 2018): The Stress test set reveals a model's ability to reason about antonyms and numbers, reliance on spurious lexical features, and robustness to random perturbations. It is constructed based on error analysis and creating adversarial examples from MultiNLI.

- **Break** (Glockner et al., 2018): The Breaking NLI dataset is an adversarial test set for evaluating the ability of lexical inferences. For each premise in the SNLI dataset, several hypotheses are generated by replacing a single word in the premise and manually verified by crowd-sourced workers.

A.2.2 Sentiment Analysis

- **IMDb** (Maas et al., 2011): The Internet Movie Database (IMDb) dataset is a collection of movie reviews.

- **Yelp** (Asghar, 2016): The Yelp Dataset is a collection of user reviews for businesses, products, and services.

- **Human-CAD** (Kaushik et al., 2019): The Human-CAD dataset for sentiment analysis is a counterfactually augmented dataset created by employing human annotators to rewrite a subset of the IMDb dataset.

- **Contrast** (Gardner et al., 2020): Similar to Human-CAD, Contrast is also a counterfactually augmented dataset created by manually rewriting a subset of the IMDb dataset.

A.3 Experimental Details

A.3.1 Data Preprocessing

SNLI We extract the training set and validation set from the official split of SNLI while balancing the label distribution.

SST-2 We use the official split of SST-2 without further preprocessing.

Dataset	Train	Dev
SNLI	20,000	2,400
SST-2	8,544	1,101

Table 5: Data Statistics.

A.3.2 Training Details

We provide more details about the training settings of our experiment. Our codes are implemented based on Huggingface's Transformers (Wolf et al., 2020). Table 6 shows the number of parameters

for the models we used in our experiment. All experiments are carried out on a single V100 GPU (32GB). Each experiment can be completed in less than 10 hours. We use manual search to select the best hyperparameters, and the search space is presented in Table 7. Our model selection criterion is validation accuracy for the classifier and validation perplexity for the generator.

Model	**Number of Parameters**
$\text{BERT}_{\text{BASE}}$	110M
$\text{BERT}_{\text{LARGE}}$	340M
$\text{RoBERTa}_{\text{BASE}}$	125M
$\text{RoBERTa}_{\text{LARGE}}$	355M
T5_{LARGE}	770M

Table 6: Number of parameters.

Hyperparameter	**Search Space**
classifier	
Learning Rate	*choice*[1e-5, 5e-5]
Training Epoch	*choice*[5, 10, 20]
Sequence Length	*choice*[64, 128, 350]
Optimizer	AdamW
Epsilon (for AdamW)	1e-8
Weight Decay	1e-1
generator	
Learning Rate	*choice*[1e-3, 1e-4, 1e-5]
Training Epoch	*choice*[5, 10, 20]
Sequence Length	*choice*[128, 350]
Optimizer	*choice*[AdamW, Adafactor]
Epsilon (for AdamW)	1e-8
Weight Decay	1e-1
Warmup Ratio	*choice*[0, 0.01]

Table 7: Number of parameters.

A.4 Comparing AutoCAD with More Baselines

To our knowledge, there are limited automatic label-flipping baselines other than Sentiment-CAD and FlipDA, which are already presented in Table 2 and Table 4. In this section, we further compare AutoCAD with three baselines, i.e., IRM (Dranker et al., 2021), C2L (Choi et al., 2022) and sentiment style-transfer (Shen et al., 2017). We conduct experiments using $\text{BERT}_{\text{BASE}}$. Experiment results are shown in Table 8 and Table 9. AutoCAD consistently and significantly outperforms all the three baselines. For IRM, we run the hypothesis-only setting from the official implementation. In line with Dranker et al. (2021), IRM shows a large variance and does not work in natural datasets. For C2L, we empirically choose the best λ between [0.1,1.0] and find the gain is slight. In line with Sentiment-CAD, we find that the sentiment-flipped data generated by style-transferring degrades the performance.

Method	Avg.
Original	$60.18_{\pm 1.9}$
IRM (Dranker et al., 2021)	$44.62_{\pm 5.1}$
C2L (Choi et al., 2022)	$60.51_{\pm 1.1}$
AutoCAD	$\mathbf{68.31}_{\pm 0.6}$

Table 8: Comparing AutoCAD with IRM and C2L on NLI. We report the mean and the standard deviation over 5 random seeds.

Method	Avg.
Original	86.43
StyleTransfer (Shen et al., 2017)	83.96
AutoCAD	**90.00**

Table 9: Comparing AutoCAD with Sentiment Style Transfer on SST.

A.5 Exploration of Robust Training for Classifier

In our main experiment, we simply combine the data generated by AutoCAD with the original data and use cross-entropy loss to train the final classifier. Considering the generated data are not perfect and may conflict with the target labels, we wonder if adopting a finer training method can better utilize the heterogeneous data and further improve the task performance. Therefore, we additionally investigate the effectiveness of the counterfactual loss (Chang et al., 2021), which does not require a label for the counterfactual data and is proved to work in training robust image classification models on CAD. Specifically, given $(\hat{x}, \hat{y})$ as a generated counterfactual from the original example (x, y), we compare the following two losses:

$$\mathcal{L}_{ce} = -log(P(\hat{y}|\hat{x}))$$
$$\mathcal{L}_{cf} = -log(1 - P(y|\hat{x}))$$

where $\mathcal{L}_{ce}$ is the standard cross-entropy loss and $\mathcal{L}_{cf}$ is the conterfactual loss. The cross-entropy loss on (x, y) is omitted for simplicity.

Method(Train loss)	**In-Domain**	**Out-of-Domain**		**Challenge**				**Avg. (%)**
	SNLI	MNLI-m	MNLI-mm	CAD	Diagnostic	Stress	Break	
Original($\mathcal{L}_{ce}$)	84.84	63.02	63.84	61.25	50.27	54.55	69.32	60.38
AutoCAD ($\mathcal{L}_{ce}$)	**87.25**	**69.67**	**70.27**	**71.43**	**54.26**	**59.13**	**89.59**	**68.38**
AutoCAD ($\mathcal{L}_{cf}$)	85.46	65.95	67.16	66.69	51.36	55.92	77.21	64.05

Table 10: Analysis of the effect of counterfactual loss for training classification models. Experiments are conducted on $\text{BERT}_{\text{BASE}}$.

We conduct experiments on SNLI with $\text{BERT}_{\text{BASE}}$. The results are shown in Table 10. We find that while training with $\mathcal{L}_{cf}$ on data augmented by AutoCAD also brings substantial improvements, $\mathcal{L}_{ce}$ consistently and significantly outperforms $\mathcal{L}_{cf}$. We conjecture that the label noise introduced by AutoCAD is relatively low. Therefore, the cross-entropy loss, which provides a stronger supervised signal, can more fully exploit the augmented data to train a robust classification model. As we focus on automatically generating counterfactuals in this paper, we leave the exploration of how to train a better classification model with AutoCAD for future work.

A.6 Generation Examples

We present detailed cases of the counterfactually augmented data generated by AutoCAD in Table 11 and Table 12. On the one hand, AutoCAD can identify valid rationales across different tasks. On the other hand, AutoCAD can generate authentic and diverse counterfactuals that conform to the target labels. Moreover, AutoCAD does not just apply simple rules to achieve label flipping. For example, in Table 11, AutoCAD generates "*young*" against "*old*" in (b) and "*crossing a river*" against "*in the snow*" in (c) as contradictions, instead of simply generating negation words such as "*not*" or "*nobody*". Another example shown in Table 12 (b) demonstrates that AutoCAD can generate counterfactuals with implicit negative expressions such as "*the only positive*". Interestingly, we also find that AutoCAD can generate plausible counterfactuals even after misidentifying rationales (see Table 12 (c)).

	Original
Premise	A man in an orange kayak and a man in a blue kayak paddle around a beautiful lake.
Hypothesis	Two men are sitting in front of a fire.
Label	Contradiction
	AutoCAD
Premise	A man in an orange kayak and a man in a blue kayak paddle around a beautiful lake.
Hypothesis	Two men are kayaking in the middle of a lake.
Label	Entailment
Premise	A man in an orange jacket sits and a man in a blue jacket around a beautiful fire.
Hypothesis	Two men are sitting in front of a fire.
Label	Entailment
Premise	A man in an orange kayak and a man in a blue kayak paddle around a beautiful lake.
Hypothesis	Two men are kayak racing in front of a crowd.
Label	Neutral

(a)

	Original
Premise	A old woman is holding shopping bags on a busy city street.
Hypothesis	A woman on a busy street holds some shopping bags.
Label	Entailment
	AutoCAD
Premise	A blond woman is chatting on a busy city street.
Hypothesis	A woman on a busy street holds some shopping bags
Label	Neutral
Premise	A old woman is holding shopping bags on a busy city street.
Hypothesis	A young woman on a busy street holding shopping bags.
Label	Contradiction

(b)

	Original
Premise	Two hikers crossing a snowy field, with mountainous terrain behind them.
Hypothesis	Two hikers are out in the snow.
Label	Entailment
	AutoCAD
Premise	Two cyclists crossing a river, with mountainous terrain behind them.
Hypothesis	Two hikers are out in the snow.
Label	Contradiction
Premise	Two cyclists traverse a ridge, with mountainous terrain behind them.
Hypothesis	Two hikers are out in the snow.
Label	Contradiction

(c)

Table 11: Examples of original SNLI data and counterfactually augmented data generated by AutoCAD. We highlight the identified rationales in original examples and the generated spans by AutoCAD.

	Original
Text	Linklater fans , or pretentious types who want to appear avant-garde will suck up to this project ...
Label	Negative
	AutoCAD
Text	Linklater fans , or pretentious types who want to appear avant-garde will adore to this project ...
Label	Positive
Text	Linklater fans , or pretentious types who want to appear avant-garde will definitely love to this project ...
Label	Positive

(a)

	Original
Text	The best thing I can say about this film is that I can't wait to see what the director does next .
Label	Positive
	AutoCAD
Text	The only positive thing I can say about this film is that I dread to see what the director does next .
Label	Negative

(b)

	Original
Text	We have an actor who is great fun to watch performing in a film that is only mildly diverting.
Label	Negative
	AutoCAD
Text	We have an actor who is great fun to watch, even if the movie is only mildly diverting ..
Label	Positive
Text	We have an actor who is great fun to watch, despite a script is only mildly diverting ..
Label	Positive

(c)

	Original
Text	A literary detective story is still a detective story and aficionados of the whodunit won't be disappointed .
Label	Positive
	AutoCAD
Text	A well-written, but uninspired detective story is still a detective story and aficionados of the genre will be disappointed .
Label	Negative

(d)

Table 12: Examples of original SST-2 data and counterfactually augmented data generated by AutoCAD. We highlight the identified rationales in original examples and the generated spans by AutoCAD.