# Controlling Learned Effects to Reduce Spurious Correlations in Text Classifiers

**Parikshit Bansal**
Microsoft Research, India
parikshitb52@gmail.com

**Amit Sharma**
Microsoft Research, India
amshar@microsoft.com

## Abstract

To address the problem of NLP classifiers learning spurious correlations between training features and target labels, a common approach is to make the model's predictions invariant to these features. However, this can be counterproductive when the features have a *non-zero* causal effect on the target label and thus are important for prediction. Therefore, using methods from the causal inference literature, we propose an algorithm to regularize the learnt effect of the features on the model's prediction to the estimated effect of feature on label. This results in an automated augmentation method that leverages the estimated effect of a feature to appropriately change the labels for new augmented inputs. On toxicity and IMDB review datasets, the proposed algorithm minimises spurious correlations and improves the minority group (i.e., samples breaking spurious correlations) accuracy, while also improving the total accuracy compared to standard training. [1]

## 1 Introduction

While classifiers trained on pre-trained NLP models achieve state-of-the-art accuracy on various tasks, they have been shown to learn spurious correlations between input features and the label (Du et al., 2022). Such learned correlations impact accuracy on out-of-distribution samples and in the case of *sensitive* spurious features, lead to unfair predictions (Sun et al., 2019; Ribeiro et al., 2020). Learned spurious correlations can be over features that are either irrelevant (e.g., tense, gender for profession classification) or relevant (e.g., emoticons for sentiment classification, negation words for contradiction). In both cases, the classifier overweighs their importance compared to other features.

For removing spurious correlations, a common principle underlying past work is to make a model's prediction *invariant* to the features that exhibit the



Figure 1: Example from IMDB reviews dataset showing the spurious token "8/10" and its importance for prediction on some inputs. Parts highlighted in yellow are **ambiguous** in sentiment, in green are (supposedly) **positive** in sentiment and red are **negative**.

correlation. This can be done by data augmentation (Kaushik et al., 2019), latent space removal (Ravfogel et al., 2020), subsampling (Sagawa et al., 2019, 2020), or sample reweighing (Mahabadi et al., 2019; Orgad and Belinkov, 2022). In many cases, however, the correlated features may be important for the task and their complete removal can cause a degradation in task performance. For instance, for spurious correlation over negation tokens (e.g., "not") or lexical overlap in MNLI natural language inference tasks, Williams et al. (2017); Joshi et al. (2022) show that correlated features are necessary for prediction and their removal can hurt accuracy.

As another example, consider the IMDB review dataset (Maas et al., 2011) where the task is classify the sentiment of a given review as positive or negative. Reviewers often include a numeric rating in their text reviews, e.g., "9/10" or "1/10". The numeric rating is highly correlated with the sentiment label, often regarded as a spurious correlation (Pezeshkpour et al., 2021) that a model should not rely on. In the first review of Fig. 1, for instance, the positive rating can mislead a classifier since the review is overall negative. However, in the second example, the text is ambiguous and the rating "8/10" can provide a helpful signal about the reviewer's sentiment (and removing it may decrease classifier's accuracy). Thus, there exist inputs where the rating is a helpful feature for prediction and other inputs where it can be counter-

---

[1]Code: https://github.com/pbansal5/feature-effect-augmentation

productive. This shows the trade-off between accuracy on *majority* groups, (i.e., samples where these correlations hold and constitute a majority of samples) and *minority* groups (i.e., comparatively fewer samples where these correlations break).

In this paper, we propose a general method to resolve the above trade-off: rather than always removing the effect of a feature on the model's prediction, we argue that the learned effect should be equal to the *true effect of the feature* on the output label. We define feature effect using the notion of conditional effect from the causal inference literature (Pearl, 2009): the change in the ground-truth label upon changing the feature, keeping all other input features constant. To enforce the true feature effect, we make **two contributions**:

1. Novel estimator of the effect of text features on the label that is accurate even at high levels of spurious correlation compared to past work.

2. Automated augmentation method that predicts the labels of new samples using the estimated feature effect and adds them to train data to achieve the desired learned effect in a classifier.

When combined with the standard accuracy loss over training data, the proposed method, *Feature Effect Augmentation (FEAG)*, obtains the highest overall accuracy compared to baselines while reducing the learnt spurious correlation. For our evaluation, we consider the practical goal of increasing the accuracy on the minority groups while not substantially reducing the accuracy over the majority group. On comment toxicity and IMDB review datasets, we find that existing methods tend to increase minority group accuracy but reduce overall accuracy, whereas FEAG obtains a good tradeoff. In some cases, it can obtain both higher overall accuracy and higher average group accuracy. Moreover, by making it easy to change the target feature effect to be enforced, FEAG provides an interpretable control mechanism to obtain any desired tradeoff between minority and majority group accuracy (setting the feature effect to zero, e.g., prioritizes minority group accuracy).

More generally, our work provides a viable direction for automated data augmentation. While existing work requires manual labeling of counterfactual examples for removing spurious correlation (Kaushik et al., 2019; Wu et al., 2021), our method can label new examples using estimated feature effects. We also show how estimated feature effects can be useful for other tasks, such as detecting annotator bias in a train set.

## 2 Related Work

Our work combines the debiasing NLP literature with causal effect estimation over text.

### 2.1 Estimating causal effect from text

Prior work on estimating causal effect on text is based on propensity scores, such as DragonNet (Shi et al., 2019) and follow-up work (Veitch et al., 2020; Gui and Veitch, 2022). However, propensity-based estimators are known to suffer from high variance, especially in text scenarios where overlap may be low (Gui and Veitch, 2022). We utilize a Riesz-based causal estimator (Chernozhukov et al., 2022) that has recently been shown to offer a better bias-variance tradeoff. In particular, it does not need to estimate the full propensity but rather estimates the weight for each sample directly, thus avoiding the variance issues of prior methods.

### 2.2 Removing spurious correlations

**Latent Space Removal.** These methods aim to remove the spurious feature from model's learnt representation. INLP (Ravfogel et al., 2020) removes spurious features by iteratively projecting learnt representations of the classifiers onto the null-space of the target class predictor. RLACE (Ravfogel et al., 2022) models the objective instead as a constrained minimax game. However, recent work shows that spurious correlations are closely entangled with rest of the sentence representation (Kumar et al., 2022; He et al., 2022), hence latent space removal methods often unintentionally remove task critical information too, leading to a degradation in model's performance.

**Weighting Methods.** Debiased Focal Loss (DFL) & Product of Experts (PoE) (Mahabadi et al., 2019) are two methods which leverage a biased model (which relies heavily on spurious features for prediction) to aid training. Specifically DFL reweighs the samples such that samples belonging to the majority group are weighed less. PoE models the task as product of two models, where one model is limited in capacity and hence captures the spurious features, where as the other learns non-spurious features. More recent versions can work without annotations for the spurious features (Orgad and
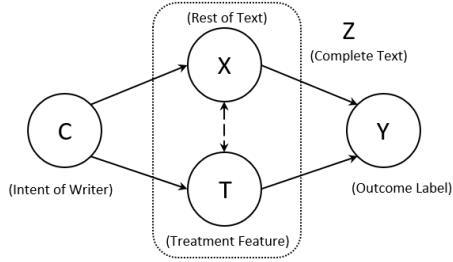
Figure 2: Causal Graph for text classification. $C$ is the intent of the writer and a hidden confounding variable. $Z$ is the complete text which is conceptually decomposed as $(X, T)$. $T$ is the treatment feature (the feature of interest) and $X$ is rest of the text. The outcome label $Y$ depends on complete text $Z$.

Belinkov, 2022), but all methods rely on reweighing the training data.

**Counterfactual Augmentation.** These methods require collection of counterfactual labeled data that can be used to regularize a classifier (Kaushik et al., 2019; Lu et al., 2020; Gupta et al., 2022). Obtaining labels for the augmented data is often prohibitively expensive.

**Comparison to our work.** All above techniques are specific ways to *remove* the impact of a spurious feature on the classifier. In comparison, we provide a general method that allows us to *control* the learned effect of a spurious feature: one can estimate the effect of a feature on the ground-truth label (which may or may not be zero) and enforce that effect on the classifier. (He et al., 2022) make a similar argument against complete removal of spurious features in the context of gender bias and rationale-based methods, while we focus on general spurious correlations and general NLP classifiers. (Joshi et al., 2022) characterise spurious correlations by necessity and sufficiency and argue for a more finegrained treatment of spurious features. In terms of implementation, our method can be seen as an extension to the counterfactual augmentation method where we automatically infer the labels for new inputs based on the modified feature's causal effect.

## 3 Estimating feature effects on labels

Our task is to estimate the effect of text features on the label $Y$ in training dataset. This is important for many use cases : **1)** regularising a text classifier to obey the feature's effect on the label in its prediction; **2)** identifying annotator artifacts (Sap et al., 2021) for the label $Y$ in the dataset, e.g., when the

estimated effect does not match the ground-truth known effect of a feature. For **1)**, we present an automated augmentation algorithm in Sec 4 based on the estimated feature effect. For **2)**, we use the feature effect estimation technique and present results on a comment toxicity dataset in Sec 5.4.

For feature effect estimation, we assume that the data is generated from a distribution $\mathcal{D}$ following the causal graph in Fig. 2 (Joshi et al., 2022; Gui and Veitch, 2022). The writer has some intent $C$, which generates the input sentence $(Z)$. The sentence $Z$ can conceptually be disentangled into 2 parts, 1) the feature of interest $(T \in \{0, 1\})$ and 2) rest of the text $X$. Annotators perceive the outcome label $(Y)$ from the complete text $Z$. The samples $\{(Z_i, Y_i)\}$ are drawn independently from $\mathcal{D}$. Note that the same dataset may contain multiple features $T^j$ $(j = 1...m)$ whose effect needs to be estimated, leading to a different decompositions $(X^j, T^j)$.

We term the feature $T$ as *treatment*, and $X$ as *covariates*, following the causality literature. Since the variables $X$ and $T$ are sampled from the same latent variable $C$, they are not independent of each other. For example, in context of IMDB data, if the intent of the writer is to write a positive review then it is highly likely that $X$ will contain positive adjectives while treatment $T$ might be the inclusion of rating as the string 9/10. This unobserved latent variable (intent of writer) is called the *confounder* $C$. The correlations between treatment feature $T$ and rest of text $X$ due to the presence of confounder $C$ can lead to the classifier model learning incorrect effect for the treatment feature. For computing feature effect, we leverage the causal inference literature (Pearl, 2009; Imbens and Rubin, 2015) and estimate *Average Treatment Effect (ATE)*.

### 3.1 Background

**Definitions.** *Propensities* (Pearl, 2009) model the probability of a covariate being treated i.e. $T = 1$. They can hence be written as $\mathcal{P}(X) = P(T = 1|X)$. *Overlap* is defined as the condition when any covariate $X$ has a non-zero probability of $T = 1$ and $T = 0$ i.e. $0 < P(T|X) < 1$ for all $X$. Overlap is a necessary condition for causal effect estimation. *Counterfactual :* Given an input $Z = (X, T)$, a counterfactual input is defined as $Z^C = (X, 1 - T)$, i.e. an input with treatment flipped and rest of the inputs kept constant. The original sample is called the *factual* input.

**Average Treatment Effect (ATE).** It is defined

as the change in label $Y$ on changing treatment $T$ from $0 \rightarrow 1$ keeping everything else constant.

$$\mathbb{E}_X[Y|X, \mathrm{do}(T=1)] - \mathbb{E}_X[Y|X, \mathrm{do}(T=0)]$$

where $\mathrm{do}()$ is the do-operator (Pearl, 2009), implying an *interventional* change in treatment $T$ while the covariates $X$ are kept constant. Assume an oracle model $g_0$ for the task, defined as $g_0(X, T=t) = \mathbb{E}[Y|X, \mathrm{do}(T=t)]$. Removing the do notation, ATE estimate can succinctly be written as,

$$\mathrm{ATE} = \frac{1}{n}\sum_i \left(g_0(X_i, 1) - g_0(X_i, 0)\right) \quad (1)$$

The above equation requires access to the oracle model $g_0$ which correctly outputs the label for counterfactual inputs $Z^C$.

An alternate formulation for computing ATE utilises propensities (of treatment $T$) i.e. $\mathcal{P}_0(X_i)$ instead of the oracle model. The ATE using this formulation is $\mathbb{E}_X[\alpha_0(Z)Y]$ ($\alpha_0$ defined below in Eq 3). Hence the ATE estimate is

$$\mathrm{ATE} = \frac{1}{n}\sum_i \alpha_0(Z_i)Y_i. \quad (2)$$

where

$$\alpha_0(Z_i) = \left(\frac{T_i}{\mathcal{P}_0(X_i)} - \frac{1 - T_i}{1 - \mathcal{P}_0(X_i)}\right) \quad (3)$$

are the *multipliers* computed from propensities.

**Direct Estimate.** The simplest method for estimating the average treatment effect is by training a model $g(.)$ as an approximation of the oracle $g_0(.)$ using the loss $g = \arg\min_g \mathbb{E}_{\mathcal{D}}[\mathcal{L}(Y, g(Z))]$. The direct estimate of the ATE can then be computed by substituting $g_0(.)$ by $g(.)$ in Eqn. 1. This gives the direct estimate (Shalit et al., 2017),

$$\hat{\mathrm{ATE}}_{\mathrm{Direct}} = \frac{1}{n}\sum_i \left(g(X_i, 1) - g(X_i, 0)\right) \quad (4)$$

The problem with using the direct estimate is that, in cases where $T$ is correlated with $X$ under $\mathcal{D}$, a loss optimizing method might exploit spurious correlations between $X$ and $T$ to learn a biased model $g(.)$. That is, the model might over(or under)-estimate the effect of $T$ on the output $Y$. This leads to a biased $\hat{\mathrm{ATE}}$.

**Propensity-based Doubly Robust (DR) Estimate.** To resolve the issue of a biased model $g$, DR estimator (Kang and Schafer, 2007; Veitch et al., 2020) utilises propensities. Since the true propensities $\mathcal{P}_0$ are unknown we learn these propensities using the loss $\mathcal{P}_{\mathrm{Pr}} = \arg\min_{\mathcal{P}} \mathbb{E}_{\mathcal{D}}[\mathcal{L}(T, \mathcal{P}(X))]$ giving estimated multipliers $\alpha_{\mathrm{Pr}}(Z_i)$.

$$\hat{\mathrm{ATE}}_{\mathrm{DR,Pr}} = \hat{\mathrm{ATE}}_{\mathrm{Direct}} + \frac{1}{n}\sum_i \alpha_{\mathrm{Pr}}(Z_i)(Y_i - g(Z_i)) \quad (5)$$

The DR estimator corrects the bias in $g$ using the correction term (second term in Eqn 5). If $g$ is systematically wrong on a minority group of examples, their residual error will add up in the correction term. Also, weighing by $\alpha_{\mathrm{Pr}}(Z_i)$ breaks correlation between $X$ and $T$, giving an unbiased correction.

### 3.2 Riesz Representer (RR) Estimator

While propensity-based methods are the most popular for estimating treatment effect, they suffer from high variance when $P(T = 1|X)$ is close to either 1 or 0 (Swaminathan and Joachims, 2015), due to the propensity terms in the denominator of the multipliers $\alpha_{\mathrm{Pr}}(.)$. This is especially a problem in high-dimensional text data, where given a treatment $T$ (e.g., a token) the probability of it occurring with most covariate texts $X$ may be close to 0 (e.g., if the covariate $X$ is about a happy incident, probability of a token like "kill" occurring in the sentence is near 0). Therefore, we propose a doubly robust estimator for text data based on recent work (Chernozhukov et al., 2022) that avoids estimating the propensities as an intermediate step. Instead it models the coefficient $\alpha_{\mathrm{Pr}}(Z)$ directly.

The proposed method depends on the Reisz representation theorem (Chernozhukov et al., 2018).

**Theorem** (Riesz Representer Theorem). *For a square integrable function $f(Z)$ (i.e. $\mathbb{E}[f^2(Z)] < \infty$), there exists a square integrable function $\alpha_R(Z)$ such that*

$$\mathbb{E}[m((Y, Z); f)] = \mathbb{E}[\alpha_R(Z)f(Z)]$$

*if and only if $\mathbb{E}[m((Y, Z); f)]$ is a continuous linear functional of $f$.*

Since the moment functional in ATE formulation (i.e. $m((Y, Z); f) = f(X, 1) - f(X, 0)$) is indeed a continuous linear functional of $f$, Riesz theorem for our purposes can be written as :

$$\mathbb{E}[f(X, 1) - f(X, 0)] = \mathbb{E}[\alpha_R(Z)f(Z)]$$

for a square integrable function $f$. Taking $f$ as $g_0$ (assuming $g_0$ is square integrable), LHS of the equality ($\mathbb{E}[g_0(X, 1) - g_0(X, 0)]$) is exactly the ATE and the RHS ($\mathbb{E}[\alpha_R(Z)g_0(Z)]$) can be interpreted as a weighted average, as in the propensity formulation of ATE (Eqn. 2). This means that $\alpha_R$ serves as an alternative formulation for $\alpha_0$. Thus, rather than using the inverse of learnt propensities $\mathcal{P}_{Pr}$ (i.e. $\alpha_{Pr}$), we can use the Riesz Representer function $\alpha_R$ as an approximation for $\alpha_0$.

The challenge now remains on how we can estimate the $\alpha_R$ function. To derive an estimation method for $\alpha_R$, we use its definition from the Riesz Representation theorem, i.e., $\alpha_R(Z)$ weighed by any bounded function $f(Z)$ gives $\mathbb{E}[f(X, 1) - f(X, 0)]$, as done by Chernozhukov et al. (2022).

$$\begin{aligned} \alpha_R &= \arg\min_\alpha \mathbb{E}[(\alpha_R(Z) - \alpha(Z))^2] \\ &= \arg\min_\alpha \mathbb{E}[\alpha_R(Z)^2 - 2\alpha_R(Z)\alpha(Z) + \alpha(Z)^2] \\ &= \arg\min_\alpha \mathbb{E}[-2\alpha_R(Z)\alpha(Z) + \alpha(Z)^2] \\ &= \arg\min_\alpha \mathbb{E}[-2(\alpha(X, 1) - \alpha(X, 0)) + \alpha(Z)^2] \end{aligned}$$

The first step is a trivial equality, which says that $\alpha_R$ is the solution for the equation $\arg\min_\alpha \mathbb{E}[(\alpha_R(Z) - \alpha(Z))^2]$. In the third step, $\alpha_R(Z)^2$ can be ignored as the minimization is over $\alpha$ and then we use the Riesz Representation theorem to expand the term $\mathbb{E}[\alpha_R(Z)\alpha(Z)]$ as $\mathbb{E}[\alpha(X, 1) - \alpha(X, 0)]$, thus getting rid of $\alpha_R$ and providing an optimization objective.

The new learnt riesz function $\alpha_R$ can then be used for computing our Doubly Robust estimate. We can simply substitute $\alpha_{Pr}$ in the DR estimate Eqn 5 by $\alpha_R$, giving us RR-based $\hat{\text{ATE}}$,

$$\hat{\text{ATE}}_{DR,R} = \hat{\text{ATE}}_{Direct} + \frac{1}{n}\sum_i \alpha_R(Z_i)(Y_i - g(Z_i)) \quad (6)$$

# 4 Controlling learnt effects in a classifier

Armed with an estimator of feature effect on the label, we now describe methods to enforce the feature effect on a predictive model's output. Given data $\{(Z, Y)\}$ where $Z$ are input sentences and $Y$ is output label, the goal is to learn a predictive model $f$ for $Y$ such that the causal effect of a feature on $f(Z)$ is the same as the true feature effect, $\tau^j$ for the $j$th feature. That is, $\tau^j$ should be equal to $\mathbb{E}_{\mathcal{D}}[f(X^j, T^j = 1) - f(X^j, T^j = 0)]$ where

$X^j$ refers to all input features except $T^j$ and the expectation is over the training distribution. As discussed in Section 3, the ideal predictive function is $g_0$ since it will ensure the correct feature effect, $\tau^j = \mathbb{E}_{\mathcal{D}}[g_0(X^j, T^j = 1) - g_0(X^j, T^j = 0)]$, and will also provide high accuracy since it is the true data generating function.

## 4.1 Counterfactual-based Regularisation

To approximate the oracle function $g_0(Z)$, for a given loss $\mathcal{L}$, Standard ERM loss minimisation optimizes, $\arg\min_f \mathbb{E}_{\mathcal{D}}[\mathcal{L}(Y, f(Z))]$. But machine learning data is often *underspecified* (D'Amour et al., 2020; Lee et al., 2022), leading to the ERM returning multiple solutions $f$ with similar accuracy on validation set. These different solution $f$ weigh different features in input text differently. As a result, the obtained solution can be far from $g_0$.

Therefore, we use the provided feature effect to constraint the solution space. A first idea is to add a regularization term that aligns the model's learnt feature effect with the provided effect. Suppose that we are given a list of $m$ binary features $\{T^j\}_{1...m}$ which are suspected to have a spurious correlation (e.g., such features can be discovered using explanation methods on an ERM model (Wang et al., 2021)). We can conceptually decompose an input sentence $Z$ into $m$ different pairs $\{(X^j, T^j)\}_{1...m}$, where $X^j$ is the part of the sentence $Z$ apart from $T^j$. Then using the given feature effect $\{\tau^j\}_{1...m}$ for each feature, we can write the regularized loss,

$$\mathcal{L} + \lambda \frac{1}{m}\sum_j (f(X^j, 1) - f(X^j, 0) - \tau^j)^2 \quad (7)$$

where $\lambda$ is the regularisation constant.

While we proposed regularizing to $\tau^j$, sometimes one may want to completely remove a feature's effect based on domain knowledge. For example, a biased dataset may exhibit a non-zero feature's effect on the label, but due to fairness reasons, one would like to completely remove its effect. In that case, we can simply set $\tau^j = 0$ and apply Equation 7. When $\tau^j$ is set to zero, FEAG can be seen as optimizing the same objective as methods that aim to fully remove the feature's effect (Ravfogel et al., 2020; Mahabadi et al., 2019).

## 4.2 Augmentations for Estimated Effect

We also consider a data augmentation alternative to regularization. Given distribution $(Z, Y) \sim \mathcal{D}$, $m$ binary features $\{T^j\}_{1...m}$, and their feature effects $\{\tau^j\}_{1...m}$, we can augment along any of the

| $\tau$ | Method | DistilBERT | | | BERT | | |
|---|---|---|---|---|---|---|---|
| | | 1% Overlap | 5% Overlap | 10% Overlap | 1% Overlap | 5% Overlap | 10% Overlap |
| 0.10 | Direct | $15.23 \pm 5.50$ | $5.92 \pm 1.31$ | $0.48 \pm 1.65$ | $8.38 \pm 2.90$ | $1.80 \pm 4.66$ | $1.13 \pm 0.47$ |
| | Propensity | $\mathbf{5.81 \pm 2.76}$ | $9.80 \pm 1.52$ | $6.59 \pm 0.48$ | $8.53 \pm 3.77$ | $9.83 \pm 5.30$ | $6.01 \pm 1.04$ |
| | Riesz | $\mathbf{5.91 \pm 4.35}$ | $\mathbf{2.04 \pm 1.25}$ | $\mathbf{1.11 \pm 0.62}$ | $\mathbf{2.68 \pm 1.24}$ | $2.61 \pm 0.24$ | $\mathbf{0.88 \pm 0.74}$ |
| 0.30 | Direct | $18.79 \pm 6.36$ | $13.86 \pm 4.64$ | $5.94 \pm 0.83$ | $22.06 \pm 10.20$ | $4.38 \pm 4.77$ | $4.72 \pm 5.74$ |
| | Propensity | $23.48 \pm 2.70$ | $20.48 \pm 0.45$ | $10.23 \pm 1.19$ | $29.02 \pm 5.99$ | $23.57 \pm 4.04$ | $9.61 \pm 2.79$ |
| | Riesz | $16.45 \pm 2.17$ | $\mathbf{0.21 \pm 1.89}$ | $\mathbf{1.45 \pm 0.22}$ | $\mathbf{0.62 \pm 5.31}$ | $\mathbf{2.92 \pm 0.81}$ | $\mathbf{2.60 \pm 1.09}$ |
| 0.50 | Direct | $16.95 \pm 3.73$ | $11.07 \pm 2.21$ | $7.51 \pm 1.56$ | $20.36 \pm 1.44$ | $17.42 \pm 1.62$ | $11.59 \pm 2.45$ |
| | Propensity | $61.88 \pm 11.10$ | $36.11 \pm 2.73$ | $17.09 \pm 1.41$ | $47.28 \pm 11.27$ | $31.41 \pm 5.72$ | $13.16 \pm 4.02$ |
| | Riesz | $15.62 \pm 3.28$ | $\mathbf{1.50 \pm 1.39}$ | $\mathbf{2.73 \pm 0.28}$ | $\mathbf{1.42 \pm 3.37}$ | $\mathbf{1.53 \pm 1.62}$ | $\mathbf{0.11 \pm 0.91}$ |

Table 1: MAE (x100) of feature effect estimate. `Riesz` gives lower error MAE than `Direct`, across values of overlap and true feature effect. `Propensity` shows high MAE error (especially in lower overlap setting).

$m$ features to generate a counterfactual distribution. When we augment along the $j$ feature, the new input becomes $Z^{j,C} = (X^j, 1 - T^j)$. Using the feature's effect $\tau^j$, we can estimate the corresponding label $Y^{j,C}$ for the input $Z^{j,C}$. Intuitively, a higher feature effect makes it more likely that the label will change (see Supp H for details). We get a new counterfactual distribution, $(Z^{j,C}, Y^{j,C}) \sim \mathcal{D}^{j,C}$.

Similarly other counterfactual distributions can be found, giving us $\{\mathcal{D}^{j,C}\}_{1...m}$. A union can be taken over these distributions to give us the counterfactual distribution over these $m$ features as $\mathcal{D}^C = \cup_{j=1}^m \mathcal{D}^{j,C}$ This new generated distribution can then be included in training as counterfactual augmentations while minimising the loss,

$$\arg \min_f \mathbb{E}_{\mathcal{D}}[\mathcal{L}(Y, f(Z))] + \lambda \mathbb{E}_{\mathcal{D}^C}[\mathcal{L}(Y, f(Z))] \quad (8)$$

where we now draw samples from the combined distribution $\mathcal{D} + \mathcal{D}^C$. $\lambda$ signifies the weighting of samples drawn from augmented counterfactual distribution $\mathcal{D}^C$ in the loss function.

While both regularisation and data augmentation can help us control the learned effect of features, owing to the scalability and ease of optimization, we use the augmentation version of our algorithm to present our results.

### 4.3 FEAG: Two-phase algorithm

To summarize, the proposed algorithm, Feature Effect Augmentation (FEAG), proceeds in two phases. It takes as input a set of features $T^j : j = 1...m$, that may be suspected to be spurious, which can be derived using an automated saliency method (e.g., top-k important tokens) (Pezeshkpour et al., 2022; Wang et al., 2021) or based on domain knowledge. **Feature effect estimation.** For each of the features $T^j$, we estimate the feature effect using the Reisz estimator from Section 3.2. We follow the 2-headed model architecture with shared parameters

| Method | BERT | | DistilBERT | |
|---|---|---|---|---|
| | CC Sub. | IMDB | CC Sub. | IMDB |
| Direct | $18.46 \pm 0.61$ | $71.93 \pm 9.36$ | $19.07 \pm 0.67$ | $66.42 \pm 9.12$ |
| Riesz | $15.77 \pm 0.50$ | $52.51 \pm 2.63$ | $15.14 \pm 0.63$ | $55.37 \pm 0.77$ |
| Propensity | $36.25 \pm 4.88$ | $45.08 \pm 10.05$ | $24.20 \pm 0.98$ | $56.86 \pm 6.75$ |

Table 2: `Riesz` estimated feature effect is less than the `Direct` learned feature effect, indicating that `Direct` method over-weighs the treatment feature.

(Shi et al., 2019) to learn the Riesz representer $\alpha_R$ and the model $g$ for $Y$ (details are in Supp J, Fig 4). Note that $\alpha_R$ and $g$ should share sentence representation extraction module to ease learning (Chernozhukov et al., 2022) (i.e., they have the same BERT model, but different final layer linear heads). These learnt models can be used in Eqn 6 to get feature effect estimates ($\{\tau^j\}_{1...m}$) on held-out data. **Counterfactual Augmentation.** Our modular pipeline allows practitioners to change the feature estimate $\tau^j$ according to their needs before using them for counterfactual augmentations. Using the features and their effect estimates, we create counterfactually augmented data $\mathcal{D}^C$ as described in Sec 4.2 and include them while training (Eqn 8) to learn the final classifier.

## 5 Experiments

We have three goals for evaluation: **1)** RR-based estimators of feature effect are more accurate than propensity-based estimators; **2)** FEAG using RR-based estimators provides better overall accuracy while minimizing spurious correlation compared to existing baselines for removing spurious correlations; **3)** Our feature effect estimator is a general method and can be used to detect annotator bias.

### 5.1 Datasets

Since the true feature effect is unknown for real-world data, we construct a semi-synthetic dataset based on the CiviComments dataset (Borkan et al.,

| S.No. | Sentence | Riesz | Interpretation |
|---|---|---|---|
| 1. | maybe it's just burning a bunch of islamist terrorists **killed** in combat | 1.69 | $T = 1$ & $P(T = 1\|X) \uparrow$ |
|  | only arabs doing what they do best, **killing** other arabs. black september, darfur, isis/isil. | 1.87 | $T = 1$ & $P(T = 1\|X) \uparrow$ |
| 2. | "strong stated desire to **kill** people in the name of islamic state" that is the important part. | 6.13 | $T = 1$ & $P(T = 1\|X) \downarrow$ |
|  | who do you think is **killing** the women its the male nativees and we hear about this | 6.14 | $T = 1$ & $P(T = 1\|X) \downarrow$ |
| 3. | they also never tell you how often the officer doing the shooting...is black | -3.75 | $T = 0$ & $P(T = 1\|X) \uparrow$ |
|  | driving into crowds of people is a popular approach for muslim terrorists | -4.53 | $T = 0$ & $P(T = 1\|X) \uparrow$ |
| 4. | i am getting very tired about hearing anything from this neurotic woman. give it a rest. | -0.56 | $T = 0$ & $P(T = 1\|X) \downarrow$ |
|  | and these men give so much to charity. there is no record of trump's charity contributions. | -0.82 | $T = 0$ & $P(T = 1\|X) \downarrow$ |

Table 3: Examples on `Kill` keyword in CivilComments Subsampled dataset. Sentences having violent words (other than `kill`) are assigned a lower score, while sentences having non-violent context are assigned high score.

2019). In addition, we evaluate on subsampled versions of the CivilComments and IMDB dataset.

**CivilComments Semi-Synthetic (SS).** CivilComments is a toxicity detection dataset $\{(X, Y)\}$, where $X$ are input sentences and $Y$ is the toxicity label (1 means *toxic*). To evaluate our methods, we need to construct a dataset generated from the causal graph in Fig. 2. Since the writer's intent (confounder) is unknown, we construct it as a property of the input text, $W = h(X) \in \{0, 1\}$, leading to the modified causal graph in Fig. 3 (Supp G). To obtain $h(X)$, we train a binary classifier using a DistilBERT model on $(X, Y)$ pairs. Finally we sample a new label as $Y' \sim \text{Bernoulli}((1 - \tau)Y + \tau T)$, giving the true feature effect as $\tau$. The complete text $Z = (X, T)$ is constructed by prepending each covariate sentence $X$ with the word `Treated` if $T = 1$ and `Untreated` if $T = 0$.

**CivilComments Subsampled.** Rather than introducing a new treatment, here we subsample CivilComments to introduce a spurious correlation between an existing token `kill` and label $Y$. Here all sentences with token `kill` are considered as treated, while others untreated. To exacerbate the spurious correlation between $T$ and $Y$, we subsample our data based on the learnt property $W$ (from above), following the causal graph in Fig 3a.

**IMDB.** From the IMDB reviews dataset (Maas et al., 2011), we consider reviews that contain a numerical rating—text string from either the set {7/,8/,9/} or {2/,3/,4/}. To construct a binary treatment variable, occurrences of these strings are replaced by `Treated` if the rating is 7, 8, or 9 and an empty string otherwise. The `Treated` token is predictive of the sentiment with 90% accuracy.

For dataset and training details, see Supp B, Supp A respectively. All results are run for 3 seeds.

## 5.2 Evaluating Feature Effect Estimation

We evaluate the performance of different estimators in Sec 3 on the CivilComments SS dataset (with different overlap $\epsilon$ and feature effects $\tau$). We

compare the `Riesz`-based DR estimator (Eqn 6) with the `Direct` (Eqn 4) and `Propensity`-based DR (Eqn 5) baselines. All estimators are finetuned using either BERT or DistilBERT as base model. See Supp **??**

**Quantitative Results.** Table 1 shows the mean error in estimating feature effect across $\tau \in \{0.10, 0.30, 0.50\}$ and $\epsilon \in \{0.01, 0.05, 0.10\}$. For hyperparameter selection, see Supp. D. Across all settings (barring 1% overlap with high $\tau$), `Riesz` is able to estimate the effect with low error. `Direct` fails to do well in high $\tau$ and low $\epsilon$ ranges, failing for both $\tau = 0.50$ and $\epsilon = 0.01$. Due to its high variance, `Propensity` is unable to work well, often producing an estimate worse than `Direct`.

For the two real-world datasets, true feature effect is unknown. But comparing the effect estimates of `Direct` and `Riesz`, `Direct` tends to overestimate the feature effect (due to spurious correlation), which is corrected to a lower value by `Riesz`.

**Qualitative Results.** To understand how the Reisz estimator works, we show qualitative results for Civil Comments Subsampled dataset in Table 3. To counter the spurious correlation of token `kill` (T) with other parts of text (X) that cause toxicity (Y), the Riesz estimator provides a low weight to sentences having features X that commonly occur with T, and higher weight to sentences having X that rarely occur with T. Treated samples (T=1) have a positive Riesz value and vice versa. We can see that sentences with violent language (in addition to `kill`) are assigned a low score while other sentences with `kill` are assigned a high score, thus serving to extract the *isolated* feature effect of `kill` (without confounding due to other tokens).

## 5.3 Accuracy of `FEAG` classifiers

We now compare `FEAG` classifiers based on `Riesz`, `FEAG(ate)`, and based on zero effect, `FEAG(0)`, with prior debiasing algorithms.

**Groups.** Classifiers that reduce spurious correlation are expected to decrease total accuracy but

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| Direct | 99.46 ± 0.08 | 3.52 ± 0.80 | 1.61 ± 0.29 | 99.42 ± 0.10 | **87.77 ± 0.02** | 51.00 ± 0.17 |
| RemoveToken | 88.71 ± 0.75 | 28.06 ± 0.94 | 37.46 ± 2.36 | 90.69 ± 0.85 | 82.80 ± 0.14 | 61.23 ± 0.45 |
| DFL | 72.45 ± 1.33 | 35.62 ± 5.51 | 53.58 ± 2.61 | 82.46 ± 3.38 | 73.45 ± 0.76 | 61.03 ± 0.77 |
| DFL-nodemog | 99.22 ± 0.34 | 4.13 ± 1.21 | 3.12 ± 0.92 | 99.34 ± 0.18 | **87.75 ± 0.10** | 51.45 ± 0.41 |
| POE | 100.00 ± 0.00 | 0.18 ± 0.14 | 0.00 ± 0.00 | 99.96 ± 0.02 | **87.94 ± 0.01** | 50.03 ± 0.03 |
| INLP | 79.10 ± 3.75 | 73.44 ± 7.52 | 38.77 ± 7.53 | 36.35 ± 9.45 | 57.54 ± 2.48 | 56.92 ± 1.41 |
| Subsample | 85.45 ± 3.98 | 59.89 ± 8.49 | 27.59 ± 8.76 | 57.72 ± 9.77 | 68.27 ± 2.54 | 57.66 ± 1.55 |
| GroupDRO | 63.98 ± 4.43 | 43.18 ± 4.68 | 59.42 ± 4.75 | 72.19 ± 3.31 | 66.02 ± 0.97 | **59.69 ± 0.28** |
| FEAG(0) | 98.89 ± 0.48 | 7.48 ± 1.77 | 4.03 ± 1.53 | 97.40 ± 0.76 | 87.01 ± 0.34 | 51.95 ± 0.31 |
| FEAG(ate) | 98.30 ± 0.30 | 4.13 ± 0.94 | 7.75 ± 1.28 | 99.36 ± 0.18 | **87.62 ± 0.06** | 52.39 ± 0.16 |

Table 4: Accuracy across groups for CivilComments Semi-Synthetic (0.50 ATE,5% Overlap), trained using BERT.

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| Direct | 76.72 ± 0.82 | 5.80 ± 1.57 | 81.72 ± 0.91 | 96.72 ± 0.35 | **79.38 ± 0.29** | 65.24 ± 0.31 |
| RemoveToken | 75.63 ± 0.79 | 15.22 ± 1.02 | 83.10 ± 0.43 | 90.15 ± 0.61 | 78.40 ± 0.23 | 66.02 ± 0.28 |
| DFL | 83.28 ± 0.16 | 9.42 ± 0.59 | 67.82 ± 0.66 | 94.09 ± 0.80 | 76.54 ± 0.36 | 63.65 ± 0.24 |
| DFL-nodemog | 78.80 ± 1.84 | 3.62 ± 1.18 | 77.82 ± 2.34 | 97.54 ± 0.46 | 78.87 ± 0.21 | 64.44 ± 0.20 |
| POE | 79.02 ± 0.62 | 10.14 ± 1.57 | 79.43 ± 0.66 | 95.24 ± 0.71 | **79.30 ± 0.37** | 65.96 ± 0.52 |
| INLP | 69.02 ± 1.04 | 6.52 ± 2.51 | 88.45 ± 0.10 | 95.07 ± 0.57 | 78.55 ± 0.34 | 64.77 ± 0.25 |
| Subsample | 73.99 ± 0.32 | 28.26 ± 2.72 | 83.45 ± 1.14 | 84.40 ± 0.97 | 77.25 ± 0.45 | **67.52 ± 0.17** |
| GroupDRO | 78.14 ± 1.32 | 44.93 ± 4.27 | 73.45 ± 5.25 | 71.92 ± 2.36 | 73.22 ± 1.79 | **67.11 ± 1.20** |
| FEAG(0) | 78.25 ± 0.45 | 11.59 ± 1.18 | 79.43 ± 0.25 | 94.25 ± 0.35 | 78.87 ± 0.14 | 65.88 ± 0.28 |
| FEAG(ate) | 78.80 ± 0.32 | 10.14 ± 0.59 | 80.34 ± 0.32 | 95.73 ± 0.35 | **79.66 ± 0.17** | 66.25 ± 0.22 |

Table 5: Accuracy across groups for CivilComments Subsampled trained using BERT model.

increase the accuracy of minority inputs that do not exhibit those correlations. To study such effects on accuracy, we divide our evaluation data into four groups: Group1 $(Y=0, T=0)$, Group2 $(Y=0, T=1)$, Group3 $(Y=1, T=0)$, Group4 $(Y=1, T=1)$. In addition, we report the average group accuracy across the four groups as a measure of debiasing/reduced spurious correlation. An ideal model should achieve both high overall accuracy and high average group accuracy, demonstrating its reduced reliance on spurious features.

**Baselines.** We consider popular baselines from prior work (Joshi et al., 2022; He et al., 2022; Orgad and Belinkov, 2022): weighting methods like DFL, DFL-nodemog, Product of Experts (Mahabadi et al., 2019; Orgad and Belinkov, 2022) and latent space removal methods like INLP (Ravfogel et al., 2020). We also include worst-group accuracy methods like GroupDRO, Subsampling (Sagawa et al., 2019, 2020) from the machine learning literature, and a baseline RemoveToken that removes the treatment feature from input (see Supp C).

**Results.** For the semi-synthetic dataset (CivilComments SS) in Table 4, FEAG(ate) increases the average group accuracy while retaining similar overall accuracy as Direct. FEAG(ate) also has bet-

ter minority group accuracy (i.e. Group2,Group3) than Direct. In comparison, FEAG(0) leads to a decrease in overall accuracy and also average group accuracy compared to FEAG(ate). Other baselines like Subsample, GroupDRO or DFL achieve a higher average group accuracy as they improve accuracy on the minority groups, but they suffer a substantial reduction in overall accuracy, from 87 to 66-73, which hinders usability of the model. Methods like DFL-nodemog or POE have no impact or obtain worse results compared to Direct. These results show the fundamental tradeoff between total and average group accuracy and how FEAG(ate) provides a good tradeoff between the two.

For the subsampled dataset (CivilComments Subsampled) in Table 5, we see a similar trend, where FEAG(ate) gives the best tradeoff between overall and average accuracy. FEAG(0) is substantially worse than FEAG(ate), showing the importance of not fully removing the effect of a spurious token. Except POE, Subsample and GroupDRO, all other methods obtain both lower total and average group accuracies compared to FEAG(ate). As before, POE is near identical to Direct while the weighting methods Subsample and GroupDRO lead to significant decreases in total accuracy.

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| `Direct` | $98.53 \pm 0.73$ | $5.82 \pm 2.16$ | $20.78 \pm 8.84$ | $99.87 \pm 0.05$ | $88.98 \pm 0.38$ | $56.25 \pm 2.25$ |
| `RemoveToken` | $81.96 \pm 1.69$ | $79.37 \pm 1.98$ | $69.26 \pm 1.77$ | $76.73 \pm 2.67$ | $78.71 \pm 0.82$ | $\mathbf{76.83 \pm 0.50}$ |
| `DFL` | $96.87 \pm 1.27$ | $8.99 \pm 6.72$ | $30.30 \pm 9.52$ | $99.28 \pm 0.51$ | $88.78 \pm 0.29$ | $58.86 \pm 3.00$ |
| `DFL-nodemog` | $94.82 \pm 0.94$ | $7.41 \pm 3.54$ | $41.56 \pm 5.34$ | $99.67 \pm 0.27$ | $88.70 \pm 0.00$ | $60.86 \pm 1.71$ |
| `POE` | $98.59 \pm 0.84$ | $14.29 \pm 8.51$ | $24.68 \pm 4.25$ | $98.82 \pm 0.97$ | $\mathbf{89.27 \pm 0.16}$ | $59.09 \pm 1.51$ |
| `INLP` | $68.33 \pm 4.57$ | $58.73 \pm 14.62$ | $49.78 \pm 6.50$ | $50.43 \pm 14.88$ | $58.82 \pm 5.45$ | $56.82 \pm 1.34$ |
| `Subsample` | $71.53 \pm 3.64$ | $65.08 \pm 1.98$ | $74.46 \pm 2.90$ | $85.67 \pm 2.94$ | $77.51 \pm 0.28$ | $74.18 \pm 0.09$ |
| `GroupDRO` | $79.40 \pm 3.67$ | $55.56 \pm 2.70$ | $67.97 \pm 1.97$ | $90.66 \pm 0.82$ | $82.25 \pm 1.34$ | $73.40 \pm 0.51$ |
| `FEAG(0)` | $94.63 \pm 0.72$ | $33.33 \pm 7.23$ | $46.75 \pm 1.84$ | $97.30 \pm 1.09$ | $\mathbf{89.33 \pm 0.15}$ | $68.00 \pm 1.65$ |
| `FEAG(ate)` | $95.46 \pm 1.27$ | $15.34 \pm 3.03$ | $43.29 \pm 5.49$ | $99.34 \pm 0.28$ | $\mathbf{89.38 \pm 0.16}$ | $63.36 \pm 1.75$ |

Table 6: IMDB dataset; models trained using BERT. `FEAG(ate)` and `FEAG(0)` achieve highest average group accuracy.

Finally, we show results for IMDB where the causal graph is unknown and our assumptions from Fig. 3a may not be valid. Nonetheless Table 6 shows that both `FEAG(ate)` and `FEAG(0)` achieve better average group accuracy with slightly better total accuracy than the `Direct` model. Other baselines follow their usual trend: ML weighting baselines (`Subsample`, `GroupDRO`) suffer reductions in total accuracy, `DFL` and `POE` methods are unable to improve average group accuracy substantially, and `INLP` is worse for both total and average group accuracy. Besides BERT, results using DistilBERT as a base model show a similar trend (Supp F). We also report `FEAG(propen)` numbers in Supp E.

### 5.4 Detecting Annotator bias

| Token | Riesz DR | $P(Y\|T)$ | Token | Riesz DR | $P(Y\|T)$ |
|---|---|---|---|---|---|
| gay | $22.30 \pm 1.03$ | 0.66 | hate | $5.81 \pm 0.21$ | 0.68 |
| racist | $14.61 \pm 0.97$ | 0.75 | you're | $1.99 \pm 0.54$ | 0.58 |
| black | $12.87 \pm 0.36$ | 0.69 | president | $0.19 \pm 0.21$ | 0.55 |
| white | $9.91 \pm 0.34$ | 0.67 | guys | $0.13 \pm 1.24$ | 0.58 |

Table 7: Tokens `racist` and `guys` show expected feature effect (1 and 0 resp.), but high feature effect for `black` and `gay` suggests annotator bias in dataset.

While we focused on the debiasing task for classifiers, our feature effect estimator is general: we apply it to detect annotator bias in the CivilComments dataset. If the true feature effect of a token is known, we can compare it to the estimated effect to detect any annotator bias in the dataset. For tokens like "racist" and "guys" where the true effect is likely to be high and zero respectively, the estimated effect confirms the prior (see Table 7). But for tokens like "gay" or "black", our method shows a significant non-zero feature effect on the label which may indicate annotator bias, as it may be known that these tokens should have a zero effect on the toxicity label. Compared to the naive conditional probability $(Y|T)$, our effect estimator can be used to provide a better sense of how impor-

tant certain keywords are for generating the output label. (e.g., "guys" obtains a zero causal effect but $P(Y|T)$ shows a substantial deviation from 0.5).

## 6   Conclusion

Rather than fully removing a feature's effect on the classifier, we presented a method for fine-grained control of the feature's effect based on causal inference. We showed how our method allows a better tradeoff between overall accuracy and accuracy over subgroups in the data. Our preliminary study on annotator bias demonstrated that our method may be useful for detecting biases in the classification label too. As future work, a natural direction is to combine these two threads and explore how we can develop methods to regularize features' effect on the debiased label, rather than the (possibly confounded) labels provided in the dataset.

**Limitations**   One major shortcoming of FEAG method is the dependency on creation of counterfactual inputs. If there is an error in counterfactual generation, we might get a wrong feature effect estimate. Thus, for simplicity, our evaluation considered tokens as features. The parallel development of counterfactual input generation methods (Wu et al., 2021; Howard et al., 2022) would hopefully ease this issue and allow FEAG to be used reliably for spurious correlations on more complex features too.

**Ethics Statement**   This project aims to check when methods are using spurious correlation. Identification of these spurious correlation is important for debiasing i.e. removal of dependence of the model on these correlations. Our work shows how instead of complete removal of these spurious features, regularising them might be better. At the same time, this is early research work and shouldn't be used in real-world systems without further evaluation.

# References

Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2019. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of The 2019 World Wide Web Conference*.

Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. 2018. Double/debiased machine learning for treatment and structural parameters.

Victor Chernozhukov, Whitney Newey, Victor M Quintas-Martinez, and Vasilis Syrgkanis. 2022. Riesznet and forestriesz: Automatic debiased machine learning with neural nets and random forests. In *International Conference on Machine Learning*, pages 3901–3914. PMLR.

Mengnan Du, Fengxiang He, Na Zou, Dacheng Tao, and Xia Hu. 2022. Shortcut learning of large language models in natural language understanding: A survey. *arXiv preprint arXiv:2208.11857*.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D Hoffman, et al. 2020. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*.

Lin Gui and Victor Veitch. 2022. Causal estimation for text data with (apparent) overlap violations. *arXiv preprint arXiv:2210.00079*.

Umang Gupta, Jwala Dhamala, Varun Kumar, Apurv Verma, Yada Pruksachatkun, Satyapriya Krishna, Rahul Gupta, Kai-Wei Chang, Greg Ver Steeg, and Aram Galstyan. 2022. Mitigating gender bias in distilled language models via counterfactual role reversal. *arXiv preprint arXiv:2203.12574*.

Zexue He, Yu Wang, Julian McAuley, and Bodhisattwa Prasad Majumder. 2022. Controlling bias exposure for fair interpretable predictions. *arXiv preprint arXiv:2210.07455*.

Phillip Howard, Gadi Singer, Vasudev Lal, Yejin Choi, and Swabha Swayamdipta. 2022. Neurocounterfactuals: Beyond minimal-edit counterfactuals for richer data augmentation. *arXiv preprint arXiv:2210.12365*.

Guido W Imbens and Donald B Rubin. 2015. *Causal inference in statistics, social, and biomedical sciences*. Cambridge University Press.

Nitish Joshi, Xiang Pan, and He He. 2022. Are all spurious features in natural language alike? an analysis through a causal lens. *arXiv preprint arXiv:2210.14011*.

Joseph DY Kang and Joseph L Schafer. 2007. Demystifying double robustness: A comparison of alternative strategies for estimating a population mean from incomplete data. *Statistical science*, 22(4):523–539.

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*.

Abhinav Kumar, Chenhao Tan, and Amit Sharma. 2022. Probing classifiers are unreliable for concept removal and detection. *arXiv preprint arXiv:2207.04153*.

Yoonho Lee, Huaxiu Yao, and Chelsea Finn. 2022. Diversify and disambiguate: Learning from underspecified data. *arXiv preprint arXiv:2202.03418*.

Kaiji Lu, Piotr Mardziel, Fangjing Wu, Preetam Amancharla, and Anupam Datta. 2020. Gender bias in neural natural language processing. In *Logic, Language, and Security*, pages 189–202. Springer.

Andrew L. 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, Portland, Oregon, USA. Association for Computational Linguistics.

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

Hadas Orgad and Yonatan Belinkov. 2022. Debiasing nlp models without demographic information. *arXiv preprint arXiv:2212.10563*.

Judea Pearl. 2009. *Causality*. Cambridge university press.

Pouya Pezeshkpour, Sarthak Jain, Sameer Singh, and Byron Wallace. 2022. Combining feature and instance attribution to detect artifacts. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1934–1946, Dublin, Ireland. Association for Computational Linguistics.

Pouya Pezeshkpour, Sarthak Jain, Sameer Singh, and Byron C Wallace. 2021. Combining feature and instance attribution to detect artifacts. *arXiv preprint arXiv:2107.00323*.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*.

Shauli Ravfogel, Michael Twiton, Yoav Goldberg, and Ryan D Cotterell. 2022. Linear adversarial concept erasure. In *International Conference on Machine Learning*, pages 18400–18421. PMLR.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. *arXiv preprint arXiv:2005.04118*.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2019. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*.

Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. 2020. An investigation of why over-parameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR.

Maarten Sap, Swabha Swayamdipta, Laura Vianna, Xuhui Zhou, Yejin Choi, and Noah A Smith. 2021. Annotators with attitudes: How annotator beliefs and identities bias toxic language detection. *arXiv preprint arXiv:2111.07997*.

Uri Shalit, Fredrik D Johansson, and David Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*, pages 3076–3085. PMLR.

Claudia Shi, David Blei, and Victor Veitch. 2019. Adapting neural networks for the estimation of treatment effects. *Advances in neural information processing systems*, 32.

Tony Sun, Andrew Gaut, Shirlyn Tang, Yuxin Huang, Mai ElSherief, Jieyu Zhao, Diba Mirza, Elizabeth Belding, Kai-Wei Chang, and William Yang Wang. 2019. Mitigating gender bias in natural language processing: Literature review. *arXiv preprint arXiv:1906.08976*.

Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. *advances in neural information processing systems*, 28.

Victor Veitch, Dhanya Sridhar, and David Blei. 2020. Adapting text embeddings for causal inference. In *Conference on Uncertainty in Artificial Intelligence*, pages 919–928. PMLR.

Tianlu Wang, Diyi Yang, and Xuezhi Wang. 2021. Identifying and mitigating spurious correlations for improving robustness in nlp models. *arXiv preprint arXiv:2110.07736*.

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

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*.

## A  Training Details

**Architecture**  All classification methods were trained using a single linear layer on top of BERT(/DistilBERT) [CLS] token. `Riesz` uses a common BERT model for sentence reprensentation and then uses 2 seperate linear layers for learning $\alpha_R$ and $g$ seperately.

**Seeds**  We use three seeds for our experiments. 0,11,44. All numbers are reported with mean and std errors over these three seeds.

**Optimization**  We use 1e-5 learning rate for BERT parameters and 1e-4 for the final linear layer parameters. We train with 32 batch size for all our experiments. The learning rate linearly decays over training iterations. We use Adam optimizer with 1e-2 weight decay for all methods.

**Best Model Selection**  All models are trained to completion (i.e. number of epochs specified for particular dataset). The evaluation is done after every epoch and the best model is chosen over all the epochs using the validation set.

**Loss**  Binary cross entropy loss is used for all methods.

**Tokenization**  We use the standard uncased tokenizers with max length of 256 tokens.

## B  Dataset Specific Details

For all datasets we set the number of epochs such that for all methods the validation loss has bottomed and starts increasing.

**CivilComments Semi-Synthetic**  Since CivilComments is heavily skewed towards the 0 label, we resample the dataset to create a balanced data which we use in all our experiments. Since the writer's intent (confounder) is unknown, we construct it as a property of the input text, $W = h(X) \in \{0, 1\}$, leading to the modified causal graph in Fig. 3. This property could be something simple like presence of a certain word like `police` in text or something more complex like inferred ethnicity of the writer. Rather than choosing a property manually, we train distilbert for modeling $h(.)$ for a few hundred iterations. We hence use $W = h(X)$ as the property. $h(.)$ achieves $\sim 78\%$ accuracy on the task. To ensure overlap, the treatment variable is sampled from $W$ such that $0 < P(T|X) < 1$ or equivalently $0 < P(T|W) < 1$. We do this by using $T$ equal to $W$ with $\epsilon > 0$ fraction of samples flipped. Finally we sample a new label as $Y' \sim \text{Bernoulli}((1-\tau)Y + \tau T)$, giving the true feature effect as $\tau$. The complete text $Z = (X, T)$ is

constructed by prepending each covariate sentence $X$ with the word Treated if $T = 1$ and Untreated if $T = 0$. This is true for all the experiments and datasets in our setup. This also eases counterfactual generation by just changing the prepended text from Treated to Untreated (and vice-versa). The dataset has 7K train samples and 2K test samples. We train the model for 10 epochs. For controlling learnt effect, we use 0.50 ATE and 5% overlap SS.

**CivilComments Subsampled** Since kill doesn't occur often in dataset (3%) we retain only 10% of the untreated sentences.

We subsample so as to retain only 5% of the samples having $T = 1 \& W = 0$. Samples having $T = 1, W = 1$ are untouched. Samples having $T = 0$ are subsampled by 10% (as mentioned above). Our dataset has 5K train samples and 2K test samples. We train the model for 10 epochs.

**IMDB** The dataset is subsampled to have equal number of positive and negative sentiment reviews. The Treated token is predictive of the sentiment with 90% accuracy. The test set is constructed similarly. The dataset has 1354 train samples and 1328 test samples. We train the model for 30 epochs.

## C Method Specific Details

**FEAG** We use $\lambda = 0.1$ for our feature effect augmentation, i.e. loss on augmented samples is weighed 1e-1 times the loss on original samples.

**Subsample,GroupDRO** These method considers an alternate objective of maximising worst group accuracy as a condition for learning models robust to spurious correlations. For Subsample we break the correlation between $T$ and $Y$ but maintain $P(T = 1)$ and $P(Y = 1)$ invariant (following (Joshi et al., 2022)). i.e. for an input sample $P(T = 1, Y = 1) = P(T = 1)P(Y = 1)$. For GroupDRO we sample from all the four groups (as defined in Sec 5.3) equally, i.e. $P(T = 1, Y = 1) = 0.25$. Additionally we have corresponding groups weights (following the original paper) with step size of 0.01. We use heavy regularisation of 1e-2 with Adam optimizer (regularisation of 1e-1 led to degradation in numbers).

**DFL,POE,DFL-nodemog** For training the biased/weak learner model we use TinyBERT model [2]. The optimization parameters for TinyBERT model were same as that of the main model

---

[2] https://huggingface.co/prajjwal1/bert-tiny

(described above). We observed that while DFL and POE's weak learner was able to capture the bias, DFL-nodemog struggled to learn main model's success and collapsed to constant value. For POE we use $\lambda = 1.0$, i.e. the loss minimised is $\text{CE}(f_m(X), Y) + \text{CE}(\text{Softmax}(\text{Log}(f_b(X)) + \text{Log}(f_m(X))), Y)$

**INLP** We train INLP in post-hoc fashion i.e we first train a Direct model, select the best model and then apply INLP on its representation. We take the code from the official repository [3] and run it for 100 iterations with minimum accuracy stopping criterion of 0.50. We tried RLACE algorithm too, but it yeilded similar/worse results than INLP

## D Best Propensity and Riesz Eval

**Propensity Eval** We choose $\lambda = 1.0$ as the best value from the table below.

| Dataset | $\lambda = 0.1$ | $\lambda = 1.0$ | $\lambda = 10.0$ |
|---|---|---|---|
| 1% | $15.50 \pm 0.32$ | $13.62 \pm 0.26$ | $13.08 \pm 0.31$ |
| 5% | $27.31 \pm 0.02$ | $25.29 \pm 0.26$ | $25.51 \pm 0.39$ |
| 10% | $38.97 \pm 0.19$ | $36.20 \pm 0.18$ | $36.36 \pm 0.14$ |

Table 8: Propensity validation loss for different hyperparameter $\lambda$. We choose $\lambda = 1.0$ as the best value.

**Riesz Eval** We choose $\lambda = 0.01$ as the best value from the table below.

| Dataset | $\lambda = 0.01$ | $\lambda = 0.1$ | $\lambda = 1.0$ |
|---|---|---|---|
| 1% | $-9.71 \pm 0.09$ | $-64.76 \pm 3.72$ | $-68.74 \pm 2.11$ |
| 5% | $-17.83 \pm 0.20$ | $-17.87 \pm 0.15$ | $-17.28 \pm 0.16$ |
| 10% | $-61.42 \pm 1.27$ | $-9.93 \pm 0.11$ | $-9.38 \pm 0.29$ |

Table 9: Riesz validation loss for different hyperparameter $\lambda$. We choose $\lambda = 0.01$ as the best value.

## E BERT Propensity-DR based FEAG numbers

Propensity-DR based FEAG numbers on the three datasets are given in Table 10, Table 11 and Table 12.

## F DistilBERT FEAG numbers

We also show FEAG numbers on the three datasets using DistilBERT as the model in Table 13, Table 15 and Table 14

---

[3] https://github.com/shauli-ravfogel/nullspace_projection

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|--------|--------|--------|--------|--------|-------|-----------|
| FEAG(0) | $98.89 \pm 0.48$ | $7.48 \pm 1.77$ | $4.03 \pm 1.53$ | $97.40 \pm 0.76$ | $87.01 \pm 0.34$ | $51.95 \pm 0.31$ |
| FEAG(ate) | $98.30 \pm 0.30$ | $4.13 \pm 0.94$ | $7.75 \pm 1.28$ | $99.36 \pm 0.18$ | $87.62 \pm 0.06$ | $52.39 \pm 0.16$ |
| FEAG(propen) | $100.00 \pm 0.00$ | $0.00 \pm 0.00$ | $0.00 \pm 0.00$ | $100.00 \pm 0.00$ | $87.94 \pm 0.00$ | $50.00 \pm 0.00$ |

Table 10: Civil Comments Semi-Synthetic (0.50 ATE, 5% overlap); models trained using BERT.

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|--------|--------|--------|--------|--------|-------|-----------|
| FEAG(0) | $78.25 \pm 0.45$ | $11.59 \pm 1.18$ | $79.43 \pm 0.25$ | $94.25 \pm 0.35$ | $78.87 \pm 0.14$ | $65.88 \pm 0.28$ |
| FEAG(ate) | $78.80 \pm 0.32$ | $10.14 \pm 0.59$ | $80.34 \pm 0.32$ | $95.73 \pm 0.35$ | $79.66 \pm 0.17$ | $66.25 \pm 0.22$ |
| FEAG(propen) | $77.60 \pm 1.57$ | $0.00 \pm 0.00$ | $77.93 \pm 1.57$ | $99.84 \pm 0.23$ | $78.83 \pm 0.15$ | $63.84 \pm 0.12$ |

Table 11: CivilComments Subsampled dataset; models trained using BERT.

## G  Alternative Causal Graphs

We present alternate version of the primary causal graph (Fig 2) in Fig 3

## H  Label Flipping Algorithm

Consider treatment $T$, label $Y$. The desired effect as $\tau$. WLOG we can assume $\tau > 0$ (if $\tau < 0$, then make $T' = 1 - T$ and proceed with $T'$). The new counterfacutal labels are $Y^C$ and new treatment is $T^C = 1 - T$ (we will only use $T$ and $T^C$ will implicitly be $1 - T$)

Consider probabilities as :

$$
\begin{aligned}
P(Y = 1|T = 1) &= p_1 \\
P(Y = 0|T = 1) &= 1 - p_1 \\
P(Y = 0|T = 0) &= p_2 \\
P(Y = 1|T = 0) &= 1 - p_2
\end{aligned}
\tag{9}
$$

**Going from untreated to treated**  Since $\tau > 0$, changing treatment from 0 to 1, should increase the probability of outcome label being 1 (and decrease probability of it being 0) i.e. $P(Y^C = 1|T = 0) > (Y = 1|T = 0) \& P(Y^C = 0|T = 0) < (Y = 0|T = 0)$. This can be achieved by keeping $Y^C = Y$ whenever $Y = 1$ and randomly flipping certain fraction (say $\eta$) of samples having $Y = 0$ to $Y^C = 1$ ( the other $1 - \eta$ would have $Y^C = Y = 0$) With the goal of $P(Y^C = 1|T = 0) - P(Y = 1|T = 0) = \tau$, $\eta$ can be easily computed as $\frac{\tau}{p_2}$. To verify we can compute

$$
P(Y^C = 1|T = 0) = P(Y = 1|T = 0) +
$$
$$
\eta P(Y = 0|T = 0)
$$
$$
P(Y^C = 1|T = 0) = P(Y = 1|T = 0) + (\frac{\tau}{p_2})p_2
$$
$$
P(Y^C = 1|T = 0) - P(Y = 1|T = 0) = \tau
\tag{10}
$$

**Going from treated to untreated**  Similarly we can argue that $Y^C = Y$ whenever $Y = 0$ and

randomly flipping $\frac{\tau}{p_2}$ fraction of samples having $Y = 1$ to $Y^C = 0$.

## I  Computational Budget

**GPUs used**  We run our experiments on NVIDIA RTX A6000 gpus. On an average each experiment takes 1 hour to complete.

We use the BERT-base (110 Million parameters) and DistilBERT model (55 Million parameters) for computation.

## J  Two-Head Riesz Model

Sharing parameters between classifier and Riesz estimator using a two-headed model forces the shared model (e.g. BERT) to learn representations which are important for both classifier and Riesz model. While this may cause a decrease in either model's performance, this leads to a better estimate due to reduced noise in estimation (Shi et al., 2019). We present our architecture in Fig 4

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| FEAG(0) | 94.63 ± 0.72 | 33.33 ± 7.23 | 46.75 ± 1.84 | 97.30 ± 1.09 | 89.33 ± 0.15 | 68.00 ± 1.65 |
| FEAG(ate) | 95.46 ± 1.27 | 15.34 ± 3.03 | 43.29 ± 5.49 | 99.34 ± 0.28 | 89.38 ± 0.16 | 63.36 ± 1.75 |
| FEAG(propen) | 91.68 ± 2.20 | 39.15 ± 7.14 | 57.14 ± 2.81 | 96.84 ± 0.58 | 88.81 ± 0.68 | 71.21 ± 1.77 |

Table 12: IMDB dataset; models trained using BERT.

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| Direct | 99.53 ± 0.20 | 3.96 ± 1.27 | 2.62 ± 1.37 | 99.50 ± 0.14 | 87.92 ± 0.03 | 51.40 ± 0.57 |
| RemoveToken | 91.53 ± 1.20 | 26.56 ± 3.00 | 26.28 ± 2.11 | 90.50 ± 1.14 | 83.23 ± 0.09 | 58.72 ± 0.24 |
| DFL | 83.86 ± 1.75 | 49.60 ± 4.03 | 35.05 ± 3.17 | 68.01 ± 3.35 | 71.89 ± 0.75 | 59.13 ± 0.20 |
| DFL-nodemog | 99.55 ± 0.17 | 2.99 ± 1.37 | 1.81 ± 0.62 | 99.58 ± 0.16 | 87.85 ± 0.02 | 50.98 ± 0.39 |
| POE | 99.99 ± 0.01 | 0.88 ± 0.72 | 0.00 ± 0.00 | 99.81 ± 0.16 | 87.91 ± 0.02 | 50.17 ± 0.14 |
| INLP | 99.78 ± 0.18 | 99.56 ± 0.36 | 0.60 ± 0.38 | 0.60 ± 0.47 | 50.28 ± 0.13 | 50.14 ± 0.08 |
| Subsample | 74.50 ± 8.65 | 46.44 ± 12.78 | 45.52 ± 13.24 | 69.86 ± 12.15 | 69.01 ± 1.87 | 59.08 ± 1.05 |
| GroupDRO | 74.45 ± 2.92 | 65.35 ± 5.57 | 47.73 ± 5.79 | 57.52 ± 4.80 | 64.87 ± 1.20 | 61.26 ± 1.27 |
| FEAG(0) | 96.23 ± 0.13 | 13.54 ± 2.28 | 15.21 ± 0.43 | 97.11 ± 0.58 | 86.74 ± 0.08 | 55.52 ± 0.46 |
| FEAG(ate) | 99.00 ± 0.25 | 7.12 ± 0.21 | 4.93 ± 1.15 | 98.90 ± 0.05 | 87.75 ± 0.05 | 52.49 ± 0.25 |

Table 13: Accuracy across groups for CivilComments Semi-Synthetic (0.50 ATE,5% Overlap). All models are trained using DistilBERT model

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| Direct | 96.23 ± 1.95 | 22.22 ± 7.14 | 32.03 ± 6.78 | 99.21 ± 0.34 | 89.30 ± 0.53 | 62.42 ± 2.81 |
| RemoveToken | 75.30 ± 4.08 | 69.31 ± 3.77 | 74.03 ± 1.62 | 76.59 ± 2.23 | 75.46 ± 1.21 | 73.81 ± 1.13 |
| DFL | 97.57 ± 1.23 | 8.99 ± 5.52 | 26.41 ± 10.90 | 99.54 ± 0.24 | 88.96 ± 0.33 | 58.13 ± 3.39 |
| DFL-nodemog | 94.31 ± 1.39 | 28.57 ± 2.70 | 41.99 ± 3.89 | 99.21 ± 0.25 | 89.44 ± 0.43 | 66.02 ± 0.41 |
| POE | 96.29 ± 1.00 | 19.05 ± 5.85 | 38.96 ± 5.85 | 99.67 ± 0.11 | 89.81 ± 0.43 | 63.49 ± 2.31 |
| INLP | 76.90 ± 14.35 | 71.96 ± 18.57 | 31.17 ± 18.42 | 25.12 ± 18.55 | 51.14 ± 2.03 | 51.29 ± 1.03 |
| Subsample | 71.08 ± 1.47 | 68.78 ± 1.14 | 71.43 ± 1.23 | 77.65 ± 1.60 | 73.83 ± 1.34 | 72.23 ± 0.87 |
| GroupDRO | 74.98 ± 3.66 | 70.37 ± 3.12 | 73.16 ± 1.87 | 78.57 ± 2.53 | 76.17 ± 2.12 | 74.27 ± 1.00 |
| FEAG(0) | 91.94 ± 0.74 | 47.09 ± 1.14 | 55.84 ± 3.41 | 94.74 ± 0.57 | 88.36 ± 0.25 | 72.40 ± 0.76 |
| FEAG(ate) | 96.42 ± 0.42 | 30.69 ± 6.10 | 44.16 ± 2.81 | 98.09 ± 0.79 | 90.15 ± 0.07 | 67.34 ± 0.84 |

Table 14: IMDB dataset; models trained using DistilBERT

| Method | Group1 | Group2 | Group3 | Group4 | Total | Avg Group |
|---|---|---|---|---|---|---|
| Direct | 80.22 ± 0.58 | 5.80 ± 0.59 | 76.32 ± 0.47 | 97.70 ± 0.35 | 79.03 ± 0.06 | 65.01 ± 0.19 |
| RemoveToken | 76.72 ± 0.68 | 12.32 ± 0.59 | 84.02 ± 0.25 | 90.31 ± 0.97 | 78.99 ± 0.36 | 65.84 ± 0.20 |
| DFL | 85.57 ± 1.63 | 8.70 ± 2.72 | 67.01 ± 1.94 | 93.60 ± 0.70 | 76.94 ± 0.56 | 63.72 ± 0.86 |
| DFL-nodemog | 77.27 ± 3.18 | 0.00 ± 0.00 | 77.59 ± 2.54 | 98.69 ± 0.49 | 78.32 ± 0.20 | 63.39 ± 0.08 |
| POE | 81.53 ± 0.91 | 16.67 ± 2.37 | 78.74 ± 0.09 | 93.60 ± 1.53 | 79.94 ± 0.12 | 67.63 ± 0.45 |
| INLP | 72.90 ± 1.55 | 10.87 ± 2.72 | 81.84 ± 1.08 | 91.46 ± 1.10 | 77.05 ± 0.13 | 64.27 ± 0.51 |
| Subsample | 76.61 ± 1.29 | 39.13 ± 2.05 | 81.61 ± 0.82 | 81.28 ± 1.42 | 77.41 ± 0.31 | 69.66 ± 0.40 |
| GroupDRO | 78.14 ± 0.18 | 48.55 ± 3.88 | 77.47 ± 0.77 | 74.06 ± 1.19 | 75.32 ± 0.39 | 69.55 ± 0.47 |
| FEAG(0) | 77.70 ± 1.49 | 10.14 ± 1.57 | 78.62 ± 1.17 | 94.91 ± 0.94 | 78.48 ± 0.09 | 65.35 ± 0.25 |
| FEAG(ate) | 79.13 ± 0.85 | 9.52 ± 1.77 | 79.08 ± 1.32 | 96.72 ± 0.35 | 79.38 ± 0.15 | 66.36 ± 0.28 |

Table 15: Accuracy across groups for CivilComments Subsampled trained using DistilBERT model.
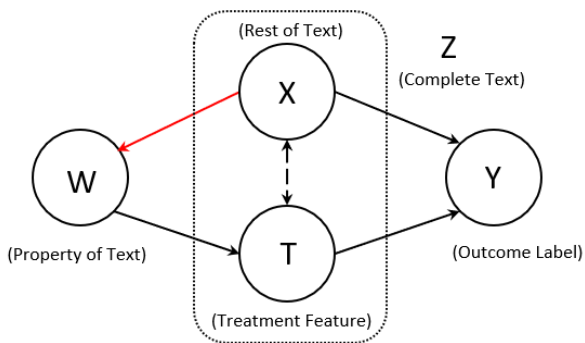
(a) Alternate Causal Graph. The Red arrow is different from Fig. 2. The confounding now instead of being the intent of writer, is a property of text $P$
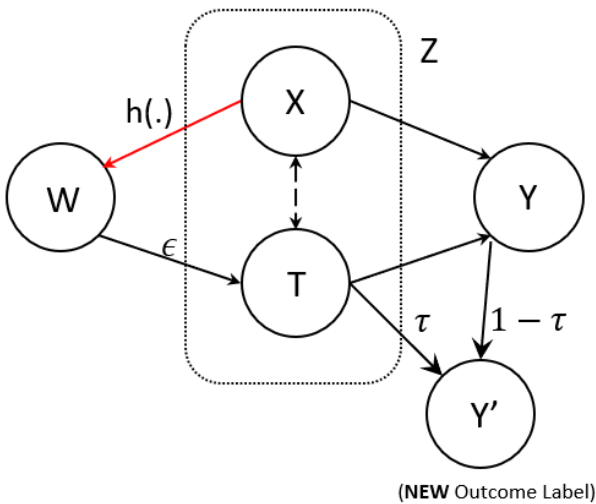


(b) Causal graph from Semi-Synthetic setting. The graph is derived from Fig. 3a, with addition of new node $Y'$. The function $h(.)$ is used to get property $W$ from $X$. Noise $\epsilon$ (in form of label flipping) is added to $W$ to ensure non-zero $P(T = t|X)$, i.e. every co-variate $X$ has non-zero probability of being treated and being untreated. $\tau$ is $T$'s effect on the new outcome label $Y'$, while $1 - \tau$ is $Y$'s affect on $Y'$
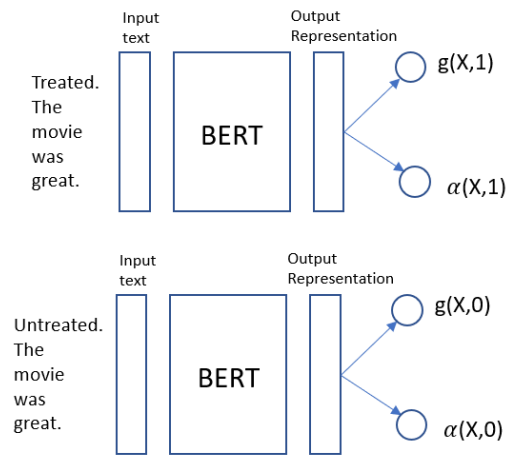
Figure 3: Alternate Causal Graphs



Figure 4: Two-head model for jointly training $\alpha_R$ and $g$ for Riesz estimator. The top figure has a treated sentence as input while the bottom figure has an untreated sentence as input.

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*Sec 6*

☑ A2. Did you discuss any potential risks of your work?
*Sec 7*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*Sec 1*

☒ A4. Have you used AI writing assistants when working on this paper?
*Left blank.*

### B ☑ Did you use or create scientific artifacts?

*Supplementary A,B,C*

☑ B1. Did you cite the creators of artifacts you used?
*Section 2,5*

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*They are all open source*

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
*Section 5*

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
*Data used doesn't contain any identifying information*

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*No response.*

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
*Supplementary B,C*

### C ☑ Did you run computational experiments?

*Sec. 5*

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
*Supplementary I*

---

*The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.*

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
*Supplementary C,D,E*

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
*Supplementary A*

☒ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
*We haven't used any packages*

**D   ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?**

*Left blank.*

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
*No response.*

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
*No response.*

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
*No response.*

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*No response.*

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*No response.*