Combating Spurious Features by Distribution Difference Regularization

Anonymous ACL submission

Abstract

Prior studies show that spurious features are inevitable to avoid in the data collection process. These spurious features cause a shortcut for a model making bad prediction in real world test data due to ignoring the real features. In this work, we focus on designing a learning scheme to hinder the model from leveraging spurious features. To achieve this, prior studies usually make strong assumptions about the spurious features and identify them purely by manipulating the training data. In contrast, we make weaker assumptions and purpose a new framework for combating spurious features by observing the distribution shift between training and auxiliary data. In particular, with the help of unlabeled auxiliary data, we design a regularization technique based on the embedding distribution difference between training and auxiliary data to mitigate the effect of spurious features. Experimental results on NLI and coreference resolution tasks demonstrate that we improve the models on out-of-domain test data and reduce the contribution of spurious features in model predictions.

1 Introduction

Recently, neural networks have demonstrated remarkable performance in several NLP benchmarks. However, due to dataset collection bias[1], several studies (Clark et al., 2019; Belinkov et al., 2019b; Sanh et al., 2021; Clark et al., 2020) show that these models may make predictions by leveraging spurious correlation between some features (a.k.a. spurious features) and class labels instead of learning to actually solve the tasks. For example, in natural language inference (NLI) datasets, SNLI (Bowman et al., 2015) and MNLI (Williams et al., 2018), a pair of sentences is more likely to be labelled as "entailment" if there are overlaps between the premise and the hypothesis, and it is more likely to be "contradiction" if the hypothesis contains negation tokens (Naik et al., 2018; McCoy et al., 2019).

Under i.i.d. assumption, when the test samples are drawn from the same distribution as the training corpus, spurious features indeed help a model in leveraging shortcut. Therefore, the model seems perform well on the benchmarks. However, when we deploy the system, the real-world might have a different distribution from the training data as they are collected from a different process. As a result, models that rely on spurious features perform terribly in the out-of-distribution samples (Mahabadi et al., 2020) as spurious features block the models from learning the correct and general features. For example, He et al. (2019); Utama et al. (2020) show that a controllable synthetic spurious feature causes the model performance drop on unbiased data even when the feature is not strong. Our goal is *is to design a learning scheme to hinder the model from leveraging spurious features during the training process.*

Prior studies on spurious features detection and mitigation often assume that the spurious features are shallow. They design models with limited capacity (e.g., linear models) to capture those features by ensemble method Clark et al. (2019); Mahabadi et al. (2020), filtering method (Sakaguchi et al., 2020; Bras et al., 2020) or adversarial training method (Belinkov et al., 2019b). However, we argue that the definition of spurious feature is not precise. In fact, not all the shallow features are spurious features. For example, in named entity recognition (NER), capitalization is a shallow feature but it is a legitimate feature that is helpful in recognizing name, locations and organizations (e.g., distinguish Apple Inc. from the fruit apple).

We argue that the key difference between spurious features and real task features is that real task features are always correlated with the task

[1] For example, the examples provided during the annotating process (Gururangan et al., 2018).

labels in a similar way, while spurious features alter when the data collecting process or distribution changes. Inspired by this, we consider spurious features as features with the following two properties: 1) spurious features are highly correlated with prediction labels. Therefore, they are often used by a model as dominant features for making predictions. 2) spurious features may not hold the same correlation with task label or may not appear in the test data if the test set is collected from a different process from the training set.

Estimating the correlation between features and task label in the training set is relatively simple as we have the task label annotations, but we do not have the distribution or task labels for the test data. Therefore, in this paper, we consider an unsupervised transfer learning setting, where we assume that a brunch of unlabelled data from an auxiliary distribution are revealed during the training. The auxiliary distribution differs from training in spurious features and it may or may not be the test distribution. We purpose an regularization by distribution difference method to control the embedding difference between training and auxiliary distribution. Discussion about the difference from unsupervised transfer learning is included in Sec. 2.

We follow the prior work to consider the NLI task with a synthetic spurious feature and the negation feature. In contrast to earilier work only consider simple text classificaiton tasks, we also consider coreference resolution, which is a more complex langauge task involved structured label which is more challenging. Results on test set where the spurious features have different distribution from training get improved, which is an evidence that the spurious features' effects are mitigated. We also do further analysis to interpret the models' behaviour in terms of those spurious features and demonstrate we indeed reduce their contribution to the final predictions.

2 Related Work

Spurious Features Prior studies show that spurious features widely exist in datasets nowadays. Among them, natural language inference (NLI) datasets are well learned, like SNLI (Bowman et al., 2015), MNLI (Williams et al., 2018) or SWAG (Zellers et al., 2018). The spurious features lay over hypothesis-only (Belinkov et al., 2019b,a), lexical features (Glockner et al., 2018; Naik et al., 2018), token overlap (McCoy et al., 2019), etc. spurious features also exist in other tasks like visual question answering (VQA) (Goyal et al., 2017), visual semantic role labeling (vSRL) (Zhao et al., 2017; Jia et al., 2020), coreference resolution (Zhao et al., 2018), etc. Models that rely on those spurious features fail to generalize to out-of-domain samples or real world scenarios (Mahabadi et al., 2020).

Following the categories defined in Shah et al. (2020), in this work we focus on the spurious features over label bias and selection bias. Compared to the prior work focusing on the the label distribution conditional on the spurious features (He et al., 2019; Mahabadi et al., 2020), we pay more attention on embedding space, which is more flexible in complex downstream tasks like structure prediction where we are not able to enumerate all possible labels. Under the selection bias umbrella, prior work usually treats spurious features as shallow but very helpful features in training set. He et al. (2019); Clark et al. (2019); Mahabadi et al. (2020) and design ensemble based methods to learn the spurious features by a shallow model and further remove it; Sakaguchi et al. (2020); Bras et al. (2020) apply adversarial filters to filter out the samples with high confidence given by shallow model to improve the out-of-domain performance. Different from these prior work, considering the spurious features are harmful in out-of-domain samples, we leverage the unlabeled out-of-domain data to help remove the spurious features.

Unsupervised Transfer Learning with Distribution Distance Regularization Utilizing distance metric between two distributions is common in unsupervised transfer learning (domain adaptation) to capture the cross-domain features to for model transfer. Metrics like Wasserstein distance (Shen et al., 2018), maximum mean discrepancy (MMD) (Long et al., 2016) and domain adversarial similarity (Ganin et al., 2016) are widely used. Although our setting and methods are similar, we are different from transfer learning in the following three aspects: 1) Our goal is different. In transfer learning we aim to better performance on the target domain, while here we focus on mitigating the effect of spurious features. Getting rid of spurious features makes models perform better in out-of-domain samples, but they are not equivalent. 2) Our motivation is different. Transfer learning is an application that leverage data from rich-resource domain to train a model on low-resource domain. However, spurious features are inevitable to avoid in the data collection process and cause a shortcut for a model making bad prediction due to ignore the real features, which is actually a systematic problem in existing machine learning models. 3) The purpose of the distribution difference regularization is different. In transfer learning usually we capture the commonalities between two diverse domains, while in our setting the train and test distribution are much more similar and we aim to get rid of the difference.

3 Regularization by Embedding Distribution Distance

According to our definition, spurious features have different behaviours in the training and auxiliary sets. Since we do not have access to the labels in auxiliary set, we estimate the spurious features based on the embedding distribution. We posit that features have similar embedding distribution in training and auxiliary are safe to use, while features with distinguishable embedding distributions should be avoided. Therefore, we design the regularization term by the embedding distribution difference between training and auxiliary .

Formally, we denote the training distribution as D_{tr} and the auxiliary distribution as D_{au}. The training set sampled from D_{tr} is denoted as $\hat{D}_{tr}$ and similarly, $\hat{D}_{au}$. In NLP tasks we usually first apply an embedding model to learn a representation for the text input, based on which we build a model to get the output for the task. We denote these two models as $E(\cdot)$, $T(\cdot)$, parameterized by θ_E, θ_T, respectively. Therefore, for a training sample $(\mathbf{x}, y)$, the loss is given by $L_{task}(T(E(\mathbf{x})), y)$, where $L_{task}(\cdot)$ is the loss function. we denote $E(D_{tr}), E(D_{au})$ as the training and auxiliary distribution in the embedding space, and the distribution distance function as $L_{dist}(\cdot)$. We define the regularized objective function as

$$L = \overbrace{\mathbb{E}_{(x,y)\sim D_{tr}}\left[L_{task}(T(E(x)), y)\right]}^{\text{Expected Loss}} + \beta \underbrace{L_{dist}(E(D_{tr}), E(D_{au}))}_{\text{Distribution Distance Regularization}} \tag{1}$$

For selection of function L_{dist}, there are numbers of functions that measure the distance between two distributions, e.g., KL-divergence, use maximum mean discrepancy (MMD) (Gretton et al., 2012), etc. In this paper we choose to compare two methods: Wasserstein distance (Arjovsky et al., 2017; Gulrajani et al., 2017; Shen et al., 2018) Jensen-Shannon (JS) divergence (Goodfellow et al., 2014) which are widely used in domain adaptation and generative adversarial networks.

3.1 Wasserstein Distance

Formally, Wasserstein distance is given by

$$W(D_1, D_2) = \sup_{\|f\|_L \leq 1} \{\mathbb{E}_{x\sim D_1}[f(x)] - \mathbb{E}_{y\sim D_2}[f(y)]\},$$

where $\|\cdot\|_L$ is the Lipschitz semi-norm, and function f is a real value function called critic function parameterized by θ_f. To remove the Lipschitz constraint, we add gradient penalty for parameter θ_f as

$$L_{grad}(x) = (\|\nabla_x f(x)\|_2 - 1)^2.$$

Empirically, let $\hat{D}_1, \hat{D}_2$ are the empirical distribution of D_1, D_2, the distance loss is given by

$$\hat{L}_{dist}(\hat{D}_1, \hat{D}_2) = -\max_{\theta_f} \left\{ \frac{1}{|\hat{D}_1|} \sum_{x\in\hat{D}_1} f(x) - \frac{1}{|\hat{D}_2|} \sum_{x\in\hat{D}_2} f(x) \right\},$$

and the function f is trained with gradient penalty-regularized distance $\hat{L}_{dist} + \lambda L_{grad}$. The training process is listed in Algorithm 1.

3.2 JS-Divergence

JS-divergence is a distance metric of two distributions, and it is used in GAN when it is firstly purposed (Goodfellow et al., 2014). However, when the support sets for the two distributions are quite different, JS-divergence suffers from gradient vanishing and cannot provide meaningful supervision. However, in our setting that training and auxiliary set are mainly from the same domain but different in spurious features, JS divergence might be more suitable. Formally, JS-divengence is defined by

$$JS(D_1, D_2) = \frac{1}{2}KL(D_1\|D_m) + \frac{1}{2}KL(D_2\|D_m),$$

where D_m is the mixture distribution as $D_m = \frac{1}{2}(D_1 + D_2)$. This value is exactly same as the cross-entropy loss of an optimal binary classifier on D_1, D_2 if we equivalently sample data from D_1 and D_2. We upper bound this divergence by

Algorithm 1 Training Process

Input: $\hat{D}_{tr}$, $\hat{D}_{au}$, L_{task}, L_{dis}, β,, scheduler $s : \mathbb{N} \to \{“task”, “adv”\}$.
Output: θ_E, θ_T, θ_f.

1: $iter \leftarrow 0$
2: $\theta_E \leftarrow$ pre-trained model
3: $\theta_T, \theta_f \leftarrow$ randomly initialize
4: **repeat**
5: sample $\mathbf{b}_{tr} = (\mathbf{x}_{tr}, \mathbf{y}_{tr})$ from $\hat{D}_{tr}$
6: sample $\mathbf{b}_{au} = \mathbf{x}_{au}$ from $\hat{D}_{au}$
7: $state \leftarrow s(iter)$
8: **if** $state == “task”$ **then**
9: $loss \leftarrow \hat{L}_{task}(\mathbf{b}_{tr}) + \beta \hat{L}_{dist}(\mathbf{b}_{tr}, \mathbf{b}_{au})$
10: $\theta_E \leftarrow \theta_E - \alpha_E \nabla_{\theta_E} loss$
11: $\theta_T \leftarrow \theta_T - \alpha_T \nabla_{\theta_T} loss.$
12: **else**
13: sample $\mathbf{x}_{grad}$ from $\mathbf{b}_{tr} \cup \mathbf{b}_{au}$
14: $\theta_f \leftarrow \theta_f +$ $\alpha_f \nabla_{\theta_f} \left(\hat{L}_{dist}(\mathbf{b}_{tr}, \mathbf{b}_{au}) + L_{grad}(\mathbf{x}_{grad}) \right)$
15: $iter \leftarrow iter + 1$
16: **until** $iter >$ MAX_ITER

the loss of a parameterized classifier f_θ. Empirically, let $\hat{D}_1, \hat{D}_2$ are the empirical distribution of D_1, D_2, we have

$$\hat{L}_{dist}(\hat{D}_1, \hat{D}_2) = - \min_{\theta_f} \left\{ \frac{1}{|\hat{D}_1|} \sum_{x \in \hat{D}_1} \log f(x) + \frac{1}{|\hat{D}_2|} \sum_{x \in \hat{D}_2} \log(1 - f(x)) \right\}$$

The training process with JS-divergence is similar as described in Algorithm 1 except that the gradient penalty part in line 14.

4 Experiments

We apply the proposed approaches in three scenarios to demonstrate its efficiency: (1) NLI task with a synthetic spurious feature; (2) NLI task with negation features; (3) coreference resolution with the gender feature. To simulate the real-world scenarios where we observe a distribution shift on some data, we create auxiliary sets based on training data manipulating the spurious feature distribution. To analyze the effect of the spurious features, we evaluate our model on a test set with different distribution of spurious features (could be different from the unlabeled auxiliary distribution). The experimental details about hyper-parameters in architectures or training are included in Appendix.

Model	Base	Ens.	Ours(JS)	Ours(W-dis)
Acc.	88.0	85.1	86.2	83.6

Table 1: The accuracy on SNLI dataset with leakage rate $p = 0$. JS stands for the JS-divergence and W-dis is the Wasserstein distance. Ens. stands for the ensemble method (He et al., 2019).

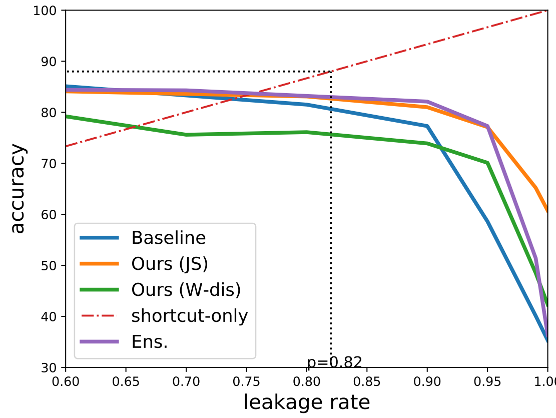

Figure 1: The accuracy on SNLI dataset with proportion of p leakage labels. Red dotted line shows that the leakage label accuracy given by $\frac{2p+1}{3}$, and $p = 0.82$ is the point that leakage label accuracy is same as the baseline performance without leakage labels.

4.1 NLI with Synthetic Leakage Label Feature

NLI is a sentence classification task on pairs of sentences. In this experiment we introduce a controllable synthetic spurious feature to demonstrate that our methods are able to work well against both strong and weak spurious features. We also compare the Wasserstein distance and JS-divergence.

Setup Following the synthetic dataset bias setting in He et al. (2019); Utama et al. (2020) on SNLI dataset, we manually leak the labels for the training and development set and concatenate it to the hypothesis. To create the auxiliary set, we equally split the original training set into two subsets and use one subset as the training set, the other as the unlabeled auxiliary set. In the training set, we leak the ground truth labels and concatenate them to the input sentence in ratio p of the instances, for the rest $(1 - p)$ of the instances this leakage label position is uniformly randomly selected. In development, auxiliary and test set the leakage label position is set as 0. Now the spurious feature (leakage label) distribution in training and the unlabeled auxiliary or test set are different. We would like to verify that this difference is able to help model avoid leveraging the spurious feature.

We use BERT (Devlin et al., 2019) as embedding and in baseline model we use a one-layer MLP to do the prediction. In our method, additionally we use a one-layer transformer (Vaswani et al., 2017) with a one-layer MLP to parameterize function f in Wasserstein distance, and use the same architecture for the discriminator. We also compare our results with ensemble method (He et al., 2019). Basically it is not exactly a fair comparison since we leverage unlabeled auxiliary data, while the ensemble method relies on then assumption that the spurious feature exists in the hypothesis.

Results The experimental results that leakage rate $p = 0$ is shown in Tab. 1, and in Fig. 1 when $p \in [0.6, 1.0]$. The performance of all models in $p \in [0, 0.6]$ are about linearly decreasing. Considering the labels in this dataset are relatively balanced, the model will get $\frac{2p+1}{3}$ accuracy if it purely relies on the leakage labels. The reference line intersects with the baseline performance ($p = 0$) at $(0.82, 0.88)$, showing that when the leakage rate $p = 0.82$, purely using the spurious feature can achieve same performance as the baseline without using it. We call the spurious feature is strong when $p > 0.82$.

When the spurious feature is not strong ($p < 0.82$) that using the spurious feature may not give better performance than using other features, we also observe the performance drop. This shows that spurious features block the model from learning the meaningful features. Although the distribution difference is not large, our method still provides supervision to reduce the spurious feature effect and our model drops less than the baseline mode. When the spurious feature gets strong, since the spurious feature is shallow, the model learns to leverage it, which gives low accuracy in test time and even random guess when $p = 1.0$. Our method shows a relatively stable performance even when in the extreme scenario. Compared to ensemble based related work (He et al., 2019) we get similar performance when the feature is not strong, while we are better in the extreme scenario where the biased model is so confident that the ensemble output may suffer from gradient vanish. This actually demonstrates that the distribution regularization provides a stronger signal to avoid leveraging the spurious feature.

Compared to the Wasserstein distance, JS-divergence is consistently better a lot. This shows that in this spurious feature mitigation scenario, the distribution difference between training and test can be better represented by JS-divergence. There are some arguments that the parameterized critic funtion in Wasserstein distance usually do not have enough capacity (Li et al., 2017). Given that we use a the multilayer transformer and a MLP layer to parameterize critic function (Wasserstein) or discriminator (JS), the capacity is more appropriate for JS. Thus, in the following experiments, we only show the results of the JS-divergence method and use "ours" to refer to it.

4.2 Coreference Resolution with Gender Feature

Prior work categorized in label bias defined in Shah et al. (2020) focuses on the imbalanced conditional distribution $p(y|h)$, where y is the task output and h is the spurious feature. This could be efficient for some tasks that the output is simple, e.g., sentence classification. However, coreference resolution task is a structure prediction task, where the output space is exponentially large. One cannot enumerate all the possibilities in the output space and spurious feature detection or mitigation is challenging.

We do experiments focusing on the gender feature stated in Zhao et al. (2018). Basically, model tends to assign higher score to male-towards occupations to pronoun he/him, or female-towards occupations to she/her due to biased data collection. To mitigate it, we may apply data argumentation (Zhao et al., 2018) method, where we can flip the gender-related tokens in training data by a rule-based approach as the argumented training set. This argumented training set can also be treated as our auxiliary set. Therefore, we would like to explore whether the effect of the gender feature can be further reduced by our method.

Setup We train our models on Ontonotes v5.0 dataset (Weischedel et al., 2013) with the argumented data. To evaluate the effect of the gender feature, we test the models on the type-2 test set in WinoBias dataset purposed in Zhao et al. (2018). In this test set, each sentence contains two occupations and one pronoun, and there is exactly one linking in ground truth from the pronoun to the second occupation, which can be correctly inferred by the grammatical structure. The test set is divided into two subsets: one is pro-stereotype

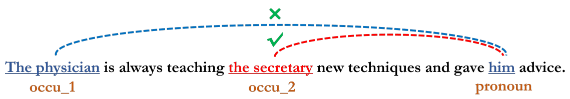

Figure 2: An example of winobias data.

Model	Test	Pro	Anti
Baseline	77.4	91.4	78.3
DA (Zhao et al., 2018)	76.9	87.9	83.8
DA+JS (Ours)	76.8	90.5	88.3

Table 2: Test is the test F1 of Ontonotes v5.0. Pro, Anti are the type-2 pro/anti-stereotype Winobias test set Zhao et al. (2018). DA stands for the data-argumentation method, and DA+JS stands for our method additionally using JS-divergence as regularization.

and one is anti-stereotype. Fig. 2 shows an example in the anti-stereotype since in training data. If the model relies much on the gender feature, the pronoun "him" tends to incorrectly link to 'physician' which is a much male-towards occupation. The absolute performance and the gap between the two test sets reflect the effect of the gender feature.

We use SpanBERT (Joshi et al., 2020) as embedding, and use the end-to-end neural architecture (Lee et al., 2017) to do the coreference resolution. For the discriminator, we use a multilayer transformer and an MLP layer to parameterize it.

Results The results are shown in Table 2. Our method further reduces the performance difference between pro-subset and anti-subset compared to purely using data argumentation. On both of the test sets our performance is better than the data argumentation, which indicates that after getting rid of the effect of the ender feature, model focuses more on the correct and meaningful features, which can be the grammatical structure in this case. However, the gender feature in the Ontonotes test set also strongly correlated to labels, we observe performance drop a little bit on the Ontonotes test set. In Sec. 5 we do experiments to further demonstrate that our model pays less attention on the gender feature.

4.3 NLI with Negation Feature

Negation feature is one of the spurious feature in NLI task (Lai and Hockenmaier, 2014; Gururangan et al., 2018; Naik et al., 2018). As shown in Naik et al. (2018), there are about 13% misclassified samples in MNLI are due to negation words and classify samples from entailment or neutral to contradiction. Thus, the model performance may drop when the distribution of the negation words changes.

Setup Since the test set of MNLI is not public, we treat the matched development set as test set, and split the training set by 90%/10% as training/development set. On development set we select the optimal hyper-parameters and use them to train the model on the whole original training set. To study the negation feature, We use pointwise mutual information (PMI) metric (Gururangan et al., 2018) to select top-5 most biased tokens towards contradiction in hypothesis: {never, no, nothing, any, none}, and treat the samples with these tokens in hypothesis as negation samples. In this experiment, we consider a even harder setting that the test data are not from the same distribution of auxiliary data. We randomly split the training set into two equal subsets D_1 and D_2 satisfying that 1) in both set the labels are balanced; 2) D_2 has no negation samples. We use D_1 to train our model while treat D_2 as the unlabeled auxiliary set. Thus D_1, D_2 are diffenent in the negation distribution. We filter out the negation samples from test set and balance the label. Thus our auxiliary set is "no-negation" set while test set is "negation-only" set. We also do testing on STRESS (Naik et al., 2018) negation test set, where each hypothesis in MNLI development set is concatenated with an adversarial suffix "and false is not true". We compare our results with the (reimplemented) ensemble based method (He et al., 2019), with the assumption that the spurious features exist in the hypothesis. The model architectures are same as Sec. 4.1.

Results The results on MNLI are shown in Table. 3. Our baseline performance and re-implementation is slightly lower than existing one with similar structure (Devlin et al., 2019) and published results, since we only use half of the training data to make the comparison to our method fair, and the distribution is also slightly different. Generally, our method improves the baseline in both neutral and contradiction classes, and keep stable in entailment in the negation only test samples. Compared to the ensemble based method, we are better in each class while the trend is similar. The reason could be that in this experiment the distribution difference mainly comes

Model	Test Acc.	Neg. Only			
		Entailment	Neutral	Contradiction	Acc.
Baseline (BERT)	81.4	92.0/74.5/82.3	80.0/61.2/69.3	67.8/96.3/79.6	77.3
Ens. (He et al., 2019)	80.9	92.0/74.5/82.3	79.4/62.0/69.6	68.3/95.9/79.8	77.5
Ours ($\beta = 2.0$)	81.2	91.0/74.5/81.9	79.0/61.2/69.4	69.3/96.3/80.6	77.6
Ours ($\beta = 5.0$)	80.8	90.6/74.9/82.0	77.3/69.0/72.9	73.6/93.8/82.5	79.3

Table 3: Performance on the MNLI matched development set. Ens. represent the ensemble method. Factor β is defined in Eq. (1). The performance for the negation-only subset per class is shown in precision / recall / F1-score, and accuracy for the rest.

Model	E	N	C
Baseline (Ours)	11.8	55.7	69.3
Ens. (He et al., 2019)	12.8	55.1	69.1
Ours ($\beta = 2.0$)	11.2	55.8	70.8
Ours ($\beta = 5.0$)	12.0	55.8	70.9

Table 4: Performance on the STRESS negation test set. E, N, C stands for entailment, neutral and contradiction, respectively. The results are shown in F1 score.

from the negation words distribution, while there can be multiple spurious features existing in the hypothesis. Our method provides a more straightforward supervision for dealing with the negation feature.

Going deep into the results, we find that there is always a relatively large gap between the precision and recall in all three classes in baseline. In contradiction class recall is greater, while in the rest the precision is greater, which verifies that model takes the spurious feature as a "prior" that samples with negation words are contradiction. Our method, in all three classes, is bridging the gap between the precision and recall, which demonstrates that we reduce this kind of "prior". Our regularization factor β provides a controllable trade-off between the overall performance and the spurious features reliance.

The results on STRESS negation test set are shown in Tab. 4. The suffix makes the sentences semantically unnatural, which can be a problem for pre-trained language models like BERT. We find that the performance for different classes are much imbalance[2]. Our methods remain stable in entailment and neutral class, and improve the baseline in contradiction, while the ensemble method has a trend to balance the classes and slightly improve the overall performance.

[2]This is even more serious in original results in He et al. (2019). Thus we reimplement it to compare with our results.

5 Analysis and Discussion

In this section we would like to demonstrate the models we learn indeed remove or mitigate the spurious features effects existing in training set. We focus on the gender feature in coreference resolution experiment and the negation feature in NLI experiment.

Coreference Resolution and Gender Bias Feature In coreference resolution we do 2-stage inference: at the first stage we do mention detection, and in the second stage we link the mention with the same reference together. Hence we focus on the linking score between the pronoun and the correct occupation in the type-2 template in WinoBias (Zhao et al., 2018) dataset. We randomly select $N = 500$ templates from the test set. For each template, we enumerate the $occu_2$ from all 40 occupations in this test set, the pronoun p from $\{her, him\}$. We denote the linking score between the pronoun and $occu_2$ in template i as $s_i(occu_2, p)$. We evaluate the gender bias (towards female) for a particular occupation o by the linking score difference to 'her' and 'him'. Considering there could also be a scaling issue in the linking sub-model, we normalize the difference by the norm of the vector of linking scores. Formally,

$$B(o) = \frac{\frac{1}{N}\sum_{i=1}^{N} s_i(o, her) - s_i(o, him)}{\sqrt{\frac{1}{2N}\sum_{i=1}^{N} s_i^2(o, her) + s_i^2(o, him)}}. \quad (2)$$

For those test case that the model fails to detect $occu_2$ as a mention candidate, we ignore this sample when we compute the average linking score.

We sort the 40 occupations by percentage of people in the occupation who are reported as female[3] and show the bias results of baseline model, data argumentation model and ours in Fig. 3.

[3]All 40 occupations and their corresponding percentage is reported in Zhao et al. (2018)

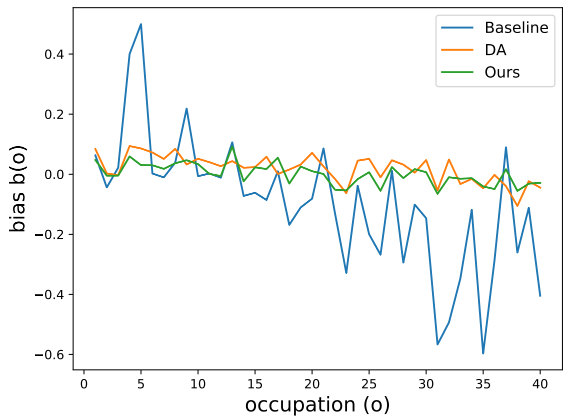

Figure 3: The bias metric for different occupations defined in Eq. (1). The occupations are sorted by the percentage of people in the occupation who are reported as female. DA stands for data argumentation method. The standard deviation of absolute bias terms of baseline, DA and Ours are 0.166, 0.0264 and 0.0208 respectively.

We know that different occupations have different frequency in the training data. For some less-frequent occupations, model does not get enough training or capacity to learn good representation for them, leading to some systematic fluctuations in the figure. We observe that the bias of the baseline model strongly correlated with the order of the occupations, while data argumentation and our method mitigate this trend. The standard deviation of $|B(o)|$ in the three curves are are 0.166, 0.0264 and 0.0208 respectively, where our method reduce the standard deviation by 21% compared to the data argumentation. Considering the existence of systematic fluctuations which we cannot remove, we believe this demonstrates that our model is further better in terms of gender fairness.

NLI and Negation Feature In NLI with negation feature we use the test performance to show that our method reduce the effect of the negation feature. To further demonstrate this, we apply LIME (Ribeiro et al., 2016) to interpret the model behaviour. Basically, LIME linearly approximates the model outputs based on pre-defined and interpretable features, and the coefficient in the linear model shows the significance of the corresponding feature. Here we use the occurrence of tokens in premises and hypothesis as binary features. We consider the top-2,000 frequent tokens in training set. Thus, for a token t, position $p \in$ {premise, hypothesis} and class $c \in$ {E, N, C}, we have feature $f_{t,p}$ and the linear regression learns a coefficient $w_{t,p,c}$ showing the contribution of token t's occurance in p to label c. The data for the linear regression are generated from the model output on MNLI matched development set, and we clip the coefficient into range $[-5, 5]$.

Model	R^2	$\bar{\lvert w \rvert}$	Avg. $w_{neg,hypo}$ E	N	C
Baseline	0.474	0.80	-1.86	-0.78	2.64
Ours	0.467	0.80	-1.57	-0.58	2.12
Diff.	-	-	-15.6%	-25.6%	-19.7%

Table 5: Results of the LIME interpretation about NLI models. Ours stands for our method when $\beta = 5.0$. R^2 is the coefficient of determination showing that how much the linear regression can represent the model. $\bar{|w|}$ is the average absolute value of all the coefficient, and Avg. $w_{neg,hypo}$ is average of coefficients about negation words in hypothesis.

The results are shown in Tab. 5. We use the coefficient of determination R^2 to show the quality of the linear regression. We believe that the value is big enough to claim the regression is meaningful. Comparing R^2 in two models, we find that our model is harder to interpret by surface features. For each class c, the average coefficient of negation tokens in hypothesis reflecting how much the model relies on this negation feature. In baseline model, the spurious feature strongly contributes to contradiction and negatively contributes to entailment. For neutral class it has negative impact but only in an average level compared to other coefficients. In our method, the scale of the coefficients and we reduce the impact of the negation bias by about 15.6% to 25.6%.

6 Conclusion

We purpose a new definition of the spurious features existing in training data with consideration about the test distribution. To mitigate their effects in machine learning models, we purpose a regularization about the distribution difference in the embedding space, which is general and can be applied in different downstream tasks. Experimental results and related analysis based on model interpretation demonstrate the effectiveness of our method in terms of spurious features mitigation. In the future, we plan to study the design and incorporation of prior knowledge from human about the spurious features.

References

Martín Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. *CoRR*, abs/1701.07875.

Yonatan Belinkov, Adam Poliak, Stuart M. Shieber, Benjamin Van Durme, and Alexander M. Rush. 2019a. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *ACL (1)*. Association for Computational Linguistics.

Yonatan Belinkov, Adam Poliak, Stuart M. Shieber, Benjamin Van Durme, and Alexander M. Rush. 2019b. On adversarial removal of hypothesis-only bias in natural language inference. In **SEM@NAACL-HLT*, pages 256–262. Association for Computational Linguistics.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*. The Association for Computational Linguistics.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew E. Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *ICML*, Proceedings of Machine Learning Research.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2019. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *EMNLP/IJCNLP (1)*, pages 4067–4080. Association for Computational Linguistics.

Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. 2020. Learning to model and ignore dataset bias with mixed capacity ensembles. In *EMNLP (Findings)*, volume EMNLP 2020 of *Findings of ACL*, pages 3031–3045. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, pages 4171–4186. Association for Computational Linguistics.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor S. Lempitsky. 2016. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI systems with sentences that require simple lexical inferences. In *ACL (2)*. Association for Computational Linguistics.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*, pages 2672–2680.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *CVPR*. IEEE Computer Society.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. 2012. A kernel two-sample test. *J. Mach. Learn. Res.*, 13:723–773.

Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. 2017. Improved training of wasserstein gans. In *NIPS*, pages 5767–5777.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL-HLT (2)*. Association for Computational Linguistics.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *DeepLo@EMNLP-IJCNLP*, pages 132–142. Association for Computational Linguistics.

Shengyu Jia, Tao Meng, Jieyu Zhao, and Kai-Wei Chang. 2020. Mitigating gender bias amplification in distribution by posterior regularization. In *ACL*, pages 2936–2942. Association for Computational Linguistics.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguistics*, 8:64–77.

Alice Lai and Julia Hockenmaier. 2014. Illinois-lh: A denotational and distributional approach to semantics. In *SemEval@COLING*, pages 329–334. The Association for Computer Linguistics.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end neural coreference resolution. In *EMNLP*, pages 188–197. Association for Computational Linguistics.

Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. 2017. MMD GAN: towards deeper understanding of moment matching network. In *NIPS*.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. 2016. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, pages 136–144.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *ACL*, pages 8706–8716. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *ACL (1)*. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman M. Sadeh, Carolyn Penstein Rosé, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *COLING*. Association for Computational Linguistics.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *KDD*, pages 1135–1144. ACM.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *AAAI*. AAAI Press.

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M. Rush. 2021. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *ICLR*. OpenReview.net.

Deven Shah, H. Andrew Schwartz, and Dirk Hovy. 2020. Predictive biases in natural language processing models: A conceptual framework and overview. In *ACL*, pages 5248–5264. Association for Computational Linguistics.

Jian Shen, Yanru Qu, Weinan Zhang, and Yong Yu. 2018. Wasserstein distance guided representation learning for domain adaptation. In *AAAI*, pages 4058–4065. AAAI Press.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Towards debiasing NLU models from unknown biases. In *EMNLP (1)*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. 2013. OntoNotes Release 5.0.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *NAACL-HLT*. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. SWAG: A large-scale adversarial dataset for grounded commonsense inference. In *EMNLP*. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2017. Men also like shopping: Reducing gender bias amplification using corpus-level constraints. In *EMNLP*. Association for Computational Linguistics.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *NAACL-HLT (2)*, pages 15–20. Association for Computational Linguistics.

A Training Details

The hyper-parameters we use are shown in Tab. 6.

Hyper-parameter	NLI-label	Coref	NLI-negation
Embedding	BERT-base	SpanBERT-base	BERT-base
#Layers in Transformer	1	1	1
#Ratio for ‘task’ state	0.2	0.4	0.2
#Layers fine-tune in BERT	6	12	6
Regularization factor	5.0	5.0	2.0/5.0
Gradient penalty factor	10.0	-	-
Warmup	8,000	14,000	14,000
Weight Decay	0.1	0.01	0.1
Optimizer	Adam	Adam	Adam
Learing Rate α_E	1e-5	2e-5	1e-5
Learing Rate α_T	3e-5	1e-4	3e-5
Learing Rate α_f	3e-5	1e-4	3e-5
Batch size	16	1	16
Epoch	8	20	20

Table 6: Hyper-parameters in our model.