End-to-End Bias Mitigation by Modelling Biases in Corpora

Rabeeh Karimi Mahabadi[♣♡] **Yonatan Belinkov**[♢] **James Henderson**[♣]
[♡]EPFL, Switzerland
[♣]Idiap Research Institute, Switzerland
[♢]Harvard University and Massachusetts Institute of Technology, Cambridge, MA, USA
{rabeeh.karimi,james.henderson}@idiap.ch
belinkov@seas.harvard.edu

Abstract

Several recent studies have shown that strong natural language understanding (NLU) models are prone to relying on unwanted dataset *biases* without learning the underlying task, resulting in models that fail to generalize to out-of-domain datasets and are likely to perform poorly in real-world scenarios. We propose two learning strategies to train neural models, which are more robust to such biases and transfer better to out-of-domain datasets. The biases are specified in terms of one or more *bias-only models*, which learn to leverage the dataset biases. During training, the bias-only models' predictions are used to adjust the loss of the base model to reduce its reliance on biases by down-weighting the biased examples and focusing training on the *hard* examples. We experiment on large-scale natural language inference and fact verification benchmarks, evaluating on out-of-domain datasets that are specifically designed to assess the robustness of models against known biases in the training data. Results show that our debiasing methods greatly improve robustness in all settings and better transfer to other textual entailment datasets. Our code and data are publicly available in https://github.com/rabeehk/robust-nli.

1 Introduction

Recent neural models (**???**) have achieved high and even near human-performance on several large-scale natural language understanding benchmarks. However, it has been demonstrated that neural models tend to rely on existing idiosyncratic biases in the datasets, and leverage superficial correlations between the label and existing shortcuts in the training dataset to perform surprisingly well,[1] without learning the underlying task (**?????**). For instance, natural language inference (NLI) is supposed to test the ability of a model to determine whether a hypothesis sentence (*There is no teacher in the room*) can be inferred from a premise sentence (*Kids work at computers with a teacher's help*) (**?**).[2] However, recent work has demonstrated that large-scale NLI benchmarks contain annotation artifacts; certain words in the hypothesis that are highly indicative of inference class and allow models that do not consider the premise to perform unexpectedly well (**??**). As an example, in some NLI benchmarks, negation words such as "nobody", "no", and "not" in the hypothesis are often highly correlated with the contradiction label.

As a result of the existence of such biases, models exploiting statistical shortcuts during training often perform poorly on out-of-domain datasets, especially if the datasets are carefully designed to limit the spurious cues. To allow proper evaluation, recent studies have tried to create new evaluation datasets that do not contain such biases (**???**). Unfortunately, it is hard to avoid spurious statistical cues in the construction of large-scale benchmarks, and collecting new datasets is costly (**?**). It is, therefore, crucial to develop techniques to reduce the reliance on biases during the training of the neural models.

We propose two end-to-end debiasing techniques that can be used when the existing bias patterns are identified. These methods work by adjusting the cross-entropy loss to reduce the biases learned from the training dataset, down-weighting the biased examples so that the model focuses on learning the hard examples. Figure **??** illustrates an example of applying our strategy to prevent an NLI model from predicting the labels using existing biases in the hypotheses, where the bias-only model only sees the hypothesis. Our strategy involves adding this bias-only branch f_B on top of the base model f_M during training. We then compute the combination of the two models f_C in a way that motivates the base model to learn different strategies than the ones used by the bias-only branch f_B. At the

[1]We use biases, heuristics or shortcuts interchangeably.

[2]The given sentences are in the contradictory relation, and the hypothesis cannot be inferred from the premise.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8706–8716
July 5 - 10, 2020.

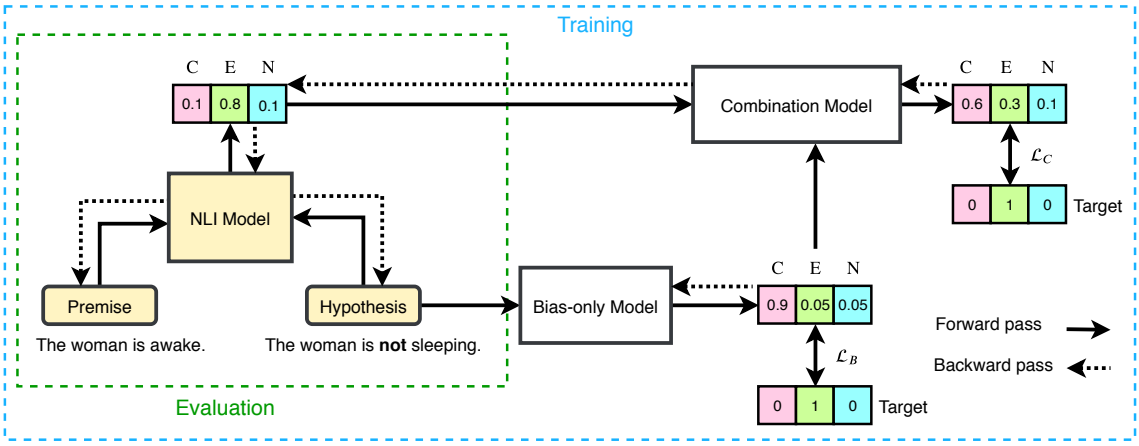

Figure 1: An illustration of our debiasing strategies applied to an NLI model. The bias-only model only sees the hypothesis, where negation words like "not" are highly correlated with the contradiction label. We train a robust NLI model by training it in combination with the bias-only model and motivate it to learn different strategies than the ones used in the bias-only model. The robust NLI model does not rely on the shortcuts and obtains improved performance on the test set.

end of the training, we remove the bias-only classifier and use the predictions of the base model.

In our first proposed method, Product of Experts, the training loss is computed on an ensemble of the base model and the bias-only model, which reduces the base model's loss for the examples that the bias-only model classifies correctly. For the second method, Debiased Focal Loss, the bias-only predictions are used to directly weight the loss of the base model, explicitly modulating the loss depending on the accuracy of the bias-only model. We also extend these methods to be robust against multiple sources of bias by training multiple bias-only models.

Our approaches are simple and highly effective. They require training only a simple model on top of the base model. They are model agnostic and general enough to be applicable for addressing common biases seen in many datasets in different domains.

We evaluate our models on challenging benchmarks in textual entailment and fact verification, including HANS (Heuristic Analysis for NLI Systems) (**?**), hard NLI sets (**?**) of Stanford Natural Language Inference (SNLI) (**?**) and MultiNLI (MNLI) (**?**), and FEVER Symmetric test set (**?**). The selected datasets are highly challenging and have been carefully designed to be unbiased to allow proper evaluation of the out-of-domain performance of the models. We additionally construct hard MNLI datasets from MNLI development sets to facilitate the out-of-domain evaluation on this dataset.[3] We show that including our strategies on training baseline models, including BERT (**?**), provides a substantial gain on out-of-domain performance in all the experiments.

In summary, we make the following contributions: 1) Proposing two debiasing strategies to train neural models robust to dataset bias. 2) An empirical evaluation of the methods on two large-scale NLI datasets and a fact verification benchmark; obtaining a substantial gain on their challenging out-of-domain data, including 7.4 points on HANS, 4.8 points on SNLI hard set, and 9.8 points on FEVER symmetric test set, setting a new state-of-the-art. 3) Proposing debiasing strategies capable of combating multiple sources of bias. 4) Evaluating the transfer performance of the debiased models on 12 NLI datasets and demonstrating improved transfer to other NLI benchmarks. To facilitate future work, we release our datasets and code.

2 Related Work

To address dataset biases, researchers have proposed to augment datasets by balancing the existing cues (**?**) or to create an adversarial dataset (**?**). However, collecting new datasets, especially at a large scale, is costly, and thus remains an unsatisfactory solution. It is, therefore, crucial to develop strategies to allow models to be trained on the existing biased datasets.

? propose to first compute the n-grams in the dataset's claims that are the most associated with each fact-verification label. They then solve an optimization problem to assign a balancing weight to each training sample to alleviate the biases. In contrast, we propose several end-to-end debiasing strategies. Additionally, **?** propose adversarial techniques to remove from the NLI sentence encoder the features that

[3] Removing the need to submit to an online evaluation system for MNLI hard test sets.

allow a hypothesis-only model to succeed. However, we believe that in general, the features used by the hypothesis-only model can include some information necessary to perform the NLI task, and removing such information from the sentence representation can hurt the performance of the full model. Their approach consequently degrades the performance on the hard SNLI set, which is expected to be less biased. In contrast, we propose to train a bias-only model to use its predictions to dynamically adapt the classification loss to reduce the importance of the most biased examples.

Concurrently to our work, **?** and **?** have also proposed to use the product of experts (PoE) models for avoiding biases. They train their models in two stages, first training a bias-only model and then using it to train a robust model. In contrast, our methods are trained in an end-to-end manner, which is convenient in practice. We additionally show that our proposed Debiased Focal Loss model is an effective method to reduce biases, sometimes superior to PoE. We have evaluated on new domains of NLI hard sets and fact verification. Moreover, we have included an analysis showing that our debiased models indeed have lower correlations with the bias-only models, and have extended our methods to guard against multiple bias patterns simultaneously. We furthermore study transfer performance to other NLI datasets.

3 Reducing Biases

Problem formulation We consider a general multi-class classification problem. Given a dataset $\mathcal{D}=\{\boldsymbol{x_i},y_i\}_{i=1}^{N}$ consisting of the input data $\boldsymbol{x_i}\in\mathcal{X}$, and labels $y_i\in\mathcal{Y}$, the goal of the base model is to learn a mapping f_M parameterized by θ_M that computes the predictions over the label space given the input data, shown as $f_M:\mathcal{X}\rightarrow\mathbb{R}^{|\mathcal{Y}|}$. Our goal is to optimize θ_M parameters such that we build a model that is more resistant to benchmark dataset biases, to improve its robustness to domain changes where the biases typically observed in the training data do not exist in the evaluation dataset.

The key idea of our approach, depicted in Figure **??**, is first to identify the dataset biases that the base model is susceptible to relying on, and define a bias-only model to capture them. We then propose two strategies to incorporate this bias-only knowledge into the training of the base model to make it robust against the biases. After training, we remove the bias-only model and use the predictions of the base model.

3.1 Bias-only Branch

We assume that we do not have access to any data from the out-of-domain dataset, so we need to know a priori about the possible types of shortcuts we would like the base model to avoid relying on. Once these patterns are identified, we train a bias-only model designed to capture the identified shortcuts that only uses *biased features*. For instance, a hypothesis-only model in the large-scale NLI datasets can correctly classify the majority of samples using annotation artifacts (**??**). Motivated by this work, our bias-only model for NLI only uses hypothesis sentences. Note that the bias-only model can, in general, have any form, and is not limited to models using only a part of the input data. For instance, on the HANS dataset, our bias-only model makes use of syntactic heuristics and similarity features (see Section **??**).

Let $\boldsymbol{x_i^b}\in\mathcal{X}^b$ be *biased features* of $\boldsymbol{x_i}$ that are predictive of y_i. We then formalize this bias-only model as a mapping $f_B:\mathcal{X}^b\rightarrow\mathbb{R}^{|\mathcal{Y}|}$, parameterized by θ_B and trained using cross-entropy (CE) loss $\mathcal{L}_B$:

$$\mathcal{L}_B(\theta_B)=-\frac{1}{N}\sum_{i=1}^{N}\log(\sigma(f_B^{y_i}(\boldsymbol{x_i^b};\theta_B))), \quad (1)$$

where $f_B^j(\boldsymbol{x_i^b},\theta_B)$ is the jth element of $f_B(.)$, and $\sigma(u^j)=e^{u^j}/\sum_{k=1}^{|\mathcal{Y}|}e^{u^k}$ is the softmax function.

3.2 Proposed Debiasing Strategies

We propose two strategies to incorporate the bias-only f_B knowledge into the training of the base model f_M. In our strategies, the predictions of the bias-only model are combined with either the predictions of the base model or its error, to down-weight the loss for the examples that the bias-only model can predict correctly. We then update parameters of the base model θ_M based on this modified loss $\mathcal{L}_C$. Our learning strategies are end-to-end. Therefore, to prevent the base model from learning the biases, the bias-only loss $\mathcal{L}_B$ is not back-propagated to any shared parameters of the base model, such as a shared sentence encoder.

3.2.1 Method 1: Product of Experts

Our first approach is based on the *product of experts* (PoE) method (**?**). Here, we use this method to combine the bias-only and base model's predictions by computing the element-wise product $\odot$ between their predictions as $\sigma(f_B(\boldsymbol{x_i^b}))\odot\sigma(f_M(\boldsymbol{x_i}))$. We compute this combination in the logarithmic space, making it appropriate for the normalized exponential below:

$$f_C(\boldsymbol{x_i},\boldsymbol{x_i^b})=\log(\sigma(f_B(\boldsymbol{x_i^b})))+\log(\sigma(f_M(\boldsymbol{x_i}))),$$

The key intuition behind this model is to combine the probability distributions of the bias-only and the base model to allow them to make predictions based on different characteristics of the input; the bias-only branch covers prediction based on biases, and the base model focuses on learning the actual task. Then the base model parameters θ_M are trained using the cross-entropy loss $\mathcal{L}_C$ of the combined classifier f_C:

$$\mathcal{L}_C(\theta_M;\theta_B) = -\frac{1}{N}\sum_{i=1}^{N}\log(\sigma(f_C^{y_i}(\boldsymbol{x_i},\boldsymbol{x_i^b}))). \quad (2)$$

When updating the base model parameters using this loss, the predictions of the bias-only model decrease the updates for examples that it can accurately predict.

Justification: Probability of label y_i for the example $\boldsymbol{x_i}$ in the PoE model is computed as:

$$\sigma(f_C^{y_i}(\boldsymbol{x_i},\boldsymbol{x_i^b})) = \frac{\sigma(f_B^{y_i}(\boldsymbol{x_i^b}))\sigma(f_M^{y_i}(\boldsymbol{x_i}))}{\sum_{k=1}^{|\mathcal{Y}|}\sigma(f_B^{k}(\boldsymbol{x_i^b}))\sigma(f_M^{k}(\boldsymbol{x_i}))}$$

Then the gradient of cross-entropy loss of the combined classifier (**??**) w.r.t θ_M is (**?**):

$$\nabla_{\theta_M}\mathcal{L}_C(\theta_M;\theta_B) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{k=1}^{|\mathcal{Y}|}\Big[\Big(\delta_{y_i k} - \sigma(f_C^k(\boldsymbol{x_i},\boldsymbol{x_i^b}))\Big)\nabla_{\theta_M}\log(\sigma(f_M^k(\boldsymbol{x_i})))\Big],$$

where $\delta_{y_i k}$ is 1 when $k{=}y_i$ and 0 otherwise. Generally, the closer the ensemble's prediction $\sigma(f_C^k(.))$ is to the target $\delta_{y_i k}$, the more the gradient is decreased through the modulating term, which only happens when the bias-only and base models are both capturing biases.

In the extreme case, when the bias-only model correctly classifies the sample, $\sigma(f_C^{y_i}(\boldsymbol{x_i},\boldsymbol{x_i^b})) = 1$ and therefore $\nabla_{\theta_M}\mathcal{L}_C(\theta_M;\theta_B) = 0$, the biased examples are ignored during training. Conversely, when the example is fully unbiased, the bias-only classifier predicts the uniform distribution over all labels $\sigma(f_B^k(\boldsymbol{x_i^b})) = \frac{1}{|\mathcal{Y}|}$ for $k \in \mathcal{Y}$, therefore $\sigma(f_C^{y_i}(\boldsymbol{x_i},\boldsymbol{x_i^b})) = \sigma(f_M^{y_i}(\boldsymbol{x_i}))$ and the gradient of ensemble classifier remains the same as the CE loss.

3.2.2 Method 2: Debiased Focal Loss

Focal loss was originally proposed in **?** to improve a single classifier by down-weighting the well-classified points. We propose a novel variant of this loss that leverages the bias-only branch's predictions to reduce the relative importance of the most biased examples and allows the model to focus on learning the *hard* examples. We define *Debiased Focal Loss* (DFL) as:

$$\mathcal{L}_C(\theta_M;\theta_B) = \quad (3)$$
$$-\frac{1}{N}\sum_{i=1}^{N}\Big(1-\sigma(f_B^{y_i}(\boldsymbol{x_i^b}))\Big)^{\gamma}\log(\sigma(f_M^{y_i}(\boldsymbol{x_i})))$$

where γ is the focusing parameter, which impacts the down-weighting rate. When γ is set to 0, DFL is equivalent to the cross-entropy loss. For $\gamma > 0$, as the value of γ is increased, the effect of down-weighting is increased. We set $\gamma = 2$ through all experiments, which works well in practice, and avoid fine-tuning it further. We note the properties of this loss: (1) When the example $\boldsymbol{x_i}$ is unbiased, and the bias-only branch does not do well, $\sigma(f_B^{y_i}(\boldsymbol{x_i^b}))$ is small, therefore the scaling factor is close to 1, and the loss remains unaffected. (2) As the sample is more biased and $\sigma(f_B^{y_i}(\boldsymbol{x_i^b}))$ is closer to 1, the modulating factor approaches 0 and the loss for the most biased examples is down-weighted.

3.3 RUBi baseline (?)

We compare our models to RUBi (**?**), a recently proposed model to alleviate unimodal biases learned by Visual Question Answering (VQA) models. **?**'s study is limited to VQA datasets. We, however, evaluate the effectiveness of their formulation on multiple challenging NLU benchmarks. RUBi consists in first applying a sigmoid function ϕ to the bias-only model's predictions to obtain a mask containing an importance weight between 0 and 1 for each label. It then computes the element-wise product between the obtained mask and the base model's predictions:

$$f_C(\boldsymbol{x_i},\boldsymbol{x_i^b}) = f_M(\boldsymbol{x_i}) \odot \phi(f_B(\boldsymbol{x_i^b})),$$

The main intuition is to dynamically adjust the predictions of the base model to prevent it from leveraging the shortcuts. Then the parameters of the base model θ_M are updated by back-propagating the cross-entropy loss $\mathcal{L}_C$ of the combined classifier.

3.4 Joint Debiasing Strategies

Neural models can, in practice, be prone to multiple types of biases in the datasets. We, therefore, propose methods for combining several bias-only models. To avoid learning relations between biased features, we do not consider training a classifier on top of their concatenation.

Instead, let $\{\boldsymbol{x_i^{b_j}}\}_{j=1}^{K}$ be different sets of *biased features* of $\boldsymbol{x_i}$ that are predictive of y_i, and let f_{B_j} be an individual bias-only model capturing $\boldsymbol{x_i^{b_j}}$. Next,

we extend our debiasing strategies to handle multiple bias patterns.

Method 1: Joint Product of Experts We extend our proposed PoE model to multiple bias-only models by computing the element-wise product between the predictions of bias-only models and the base model as: $\sigma(f_{B_1}(\boldsymbol{x_i^{b_1}})) \odot \cdots \odot \sigma(f_{B_K}(\boldsymbol{x_i^{b_K}})) \odot \sigma(f_M(\boldsymbol{x_i}))$, computed in the logarithmic space:

$$f_C(\boldsymbol{x_i}, \{\boldsymbol{x_i^{b_j}}\}_{j=1}^K) = \sum_{j=1}^{K} \log(\sigma(f_{B_j}(\boldsymbol{x_i^{b_j}}))) + \log(\sigma(f_M(\boldsymbol{x_i}))).$$

Then the base model parameters θ_M are trained using the cross-entropy loss of the combined classifier f_C.

Method 2: Joint Debiased Focal Loss To extend DFL to handle multiple bias patterns, we first compute the element-wise average of the predictions of the multiple bias-only models: $f_B(\{\boldsymbol{x_i^{b_j}}\}_{j=1}^K) = \frac{1}{K}\sum_{j=1}^K f_{B_j}(\boldsymbol{x_i^{b_j}})$, and then compute the DFL (**??**) using the computed joint bias-only model.

4 Evaluation on Unbiased Datasets

We provide experiments on a fact verification (FEVER) and two large-scale NLI datasets (SNLI and MNLI). We evaluate the models' performance on recently-proposed challenging unbiased evaluation sets. We use the BERT (**?**) implementation of **?** as our main baseline, known to work well for these tasks. In all the experiments, we use the default hyperparameters of the baselines.

4.1 Fact Verification

Dataset: The FEVER dataset contains claim-evidence pairs generated from Wikipedia. **?** collected a new evaluation set for the FEVER dataset to avoid the idiosyncrasies observed in the claims of this benchmark. They made the original claim-evidence pairs of the FEVER evaluation dataset symmetric, by augmenting them and making each claim and evidence appear with each label. Therefore, by balancing the artifacts, relying on statistical cues in claims to classify samples is equivalent to a random guess. The collected dataset is challenging, and the performance of the models relying on biases evaluated on this dataset drops significantly.

Base models: We consider BERT as the base model, which works the best on this dataset (**?**), and predicts the relations based on the concatenation of the claim and the evidence with a delimiter token (see Appendix **??**).

Bias-only model: The bias-only model predicts the labels using only claims as input.

Results: Table **??** shows the results. Our proposed debiasing methods, PoE and DFL, are highly effective, boosting the performance of the baseline by 9.8 and 7.5 points respectively, significantly surpassing the prior work of **?**.

Loss	Dev	Test	Δ
CE	85.99	56.49	
RUBi	86.23	57.60	+1.1
?	84.6	**61.6**	**+5.1**
DFL	83.07	64.02	+7.5
PoE	86.46	**66.25**	**+9.8**

Table 1: Results on FEVER development and symmetric test set. **Δ** are absolute differences with CE loss.

4.2 Natural Language Inference

Datasets: We evaluate on hard datasets of SNLI and MNLI (**?**), which are the splits of these datasets where a hypothesis-only model cannot correctly predict the labels. **?** show that the success of the recent textual entailment models is attributed to the *biased* examples, and the performance of these models is substantially lower on the *hard* sets.

Base models: We consider BERT and InferSent (**?**) as our base models. We choose InferSent to be able to compare with the prior work of **?**.

Bias-only model: The bias-only model predicts the labels using the hypothesis (Appendix **??**).

Results on SNLI: Table **??** shows the SNLI results. With InferSent, DFL and PoE result in 4.1 and 4.8 points gain. With BERT, DFL and PoE improve the results by 2.5 and 1.6 absolute points. Compared to the prior work of **?** (AdvCls), our PoE model obtains a 7.4 points gain, setting a new state-of-the-art.

Loss	BERT			InferSent		
	Test	Hard	Δ	Test	Hard	Δ
CE	90.53	80.53		84.24	68.91	
RUBi	90.69	**80.62**	**+0.1**	83.93	**69.64**	**+0.7**
AdvCls*	—	—	—	83.56	66.27	-2.6
AdvDat*	—	—	—	78.30	55.60	-13.3
DFL	89.57	**83.01**	**+2.5**	73.54	73.05	+4.1
PoE	90.11	82.15	+1.6	80.35	**73.69**	**+4.8**

Table 2: Results on the SNLI test, hard set, and differences with CE loss. *: results from **?**.

Results on MNLI: We construct hard sets from the validation sets of MNLI Matched and Mismatched (MNLI-M). Following **?**, we train a `fastText` classifier (**?**) that predicts the labels using only the hypothesis and consider the subset on which it fails as hard examples.

We report the results on MNLI mismatched sets in Table **??** (see Appendix **??** for similar results on MNLI matched). With BERT, DFL and PoE obtain 1.4 and 1.7 points gain on the hard development set, while with InferSent, they improve the results by 2.5 and 2.6 points. To comply with limited access to the MNLI submission system, we evaluate only the best result of the baselines and our models on the test sets. Our PoE model improves the performance on the hard test set by 1.1 points while retaining in-domain accuracy.

Loss	BERT			InferSent		
	MNLI	Hard	Δ	MNLI	Hard	Δ
Development set results						
CE	84.53	77.55		69.99	56.53	
RUBi	85.17	78.63	+1.1	70.53	**58.08**	**+1.5**
DFL	84.85	78.92	+1.4	61.12	59.05	+2.5
PoE	84.85	**79.23**	**+1.7**	65.85	**59.14**	**+2.6**
Test set results						
CE	83.51	75.75		—	—	—
PoE	83.47	**76.83**	**+1.1**	—	—	—

Table 3: Results on MNLI mismatched benchmark and MNLI mismatched hard set. Δ are absolute differences with CE loss.

4.3 Syntactic Bias in NLI

Dataset: **?** show that NLI models trained on MNLI can adopt superficial syntactic heuristics. They introduce HANS, consisting of several examples on which the syntactic heuristics fail.

Base model: We use BERT as our base model and train it on the MNLI dataset.

Bias-only model: We consider the following features for the bias-only model. The first four features are based on the syntactic heuristics proposed in **?**: 1) Whether all words in the hypothesis are included in the premise; 2) If the hypothesis is the contiguous subsequence of the premise; 3) If the hypothesis is a subtree in the premise's parse tree; 4) The number of tokens shared between premise and hypothesis normalized by the number of tokens in the premise. We additionally include some similarity features: 5) The cosine similarity between premise and hypothesis's pooled token representations from BERT followed by min, mean, and max-pooling. We consider the same weight for contradiction and neutral labels in the bias-only loss to allow the model to recognize entailment from not-entailment. During the evaluation, we map the neutral and contradiction labels to not-entailment.

Results: **?** observe large variability in the linguistic generalization of neural models. We, therefore, report the averaged results across 4 runs with the standard deviation in Table **??**. PoE and DFL obtain 4.4 and 7.4 points gain (see Appendix **??** for accuracy on individual heuristics of HANS).

Loss	MNLI	HANS	Δ
CE	84.51	61.88 ±1.9	
RUBi	84.53	61.76±2.7	-0.1
Reweight ❣	83.54	**69.19**	**+7.3**
Learned-Mixin ❣	84.29	64.00	+2.1
Learned-Mixin+H ❣ ✤	83.97	**66.15**	**+4.3**
PoE	84.19	66.31±0.6	+4.4
DFL	83.95	**69.26** ±0.2	**+7.4**
DFL✤	82.76	**71.95**±1.4	**+10.1**

Table 4: Results on MNLI Matched dev set and HANS. ❣: results from **?**. ✤: perform hyper-parameter tuning. Δ are differences with CE loss.

We compare our results with the concurrent work of **?**, who propose a PoE model similar to ours, which gets similar results. The main difference is that our models are trained end-to-end, which is convenient in practice, while **?**'s method requires two steps, first training a bias-only model and then using this pre-trained model to train a robust model. The Reweight baseline in **?** is a special case of our DFL with $\gamma = 1$ and performs similarly to our DFL method (using default $\gamma = 2$). Their Learned-Mixin+H method requires hyperparameter tuning. Since the assumption is

not having access to any out-of-domain test data, and there is no available dev set for HANS, it is challenging to perform hyper-parameter tuning. **?** follow prior work (**??**) and perform model section on the test set.

To provide a fair comparison, we consequently also tuned γ in DFL by sweeping over $\{0.5,1,2,3,4\}$. DFL♣ is the selected model, with $\gamma = 3$. With this hyperparameter tuning, DFL is even more effective, and our best result performs 2.8 points better than **?**.

4.4 Jointly Debiasing Multiple Bias Patterns

To evaluate combating multiple bias patterns, we jointly debias a base model on the hypothesis artifacts and syntactic biases.

Base model: We use BERT as our base model and train it on the MNLI dataset.

Loss	MNLI	Hard	Δ	HANS	Δ
CE	84.53	77.55		61.88±1.9	
PoE ♣	84.85	**79.23**	**+1.7**	60.43	-1.5
DFL♣	84.85	78.92	+1.4	60.63	-1.2
PoE ♥	84.55	77.90±0.3	+0.4	66.31±0.6	+4.4
DFL♥	84.30	77.66±0.6	+0.1	**69.26**±0.2	**+7.4**
PoE-Joint	84.39	**78.61**±0.1	**+1.1**	68.04±1.2	+6.2
DFL-Joint	84.49	78.36±0.4	+0.8	**69.10**±0.7	**+7.2**

Table 5: Results on MNLI mismatched dev set, MNLI mismatched hard set, and HANS when training independently to debias against either hypothesis artifacts (♣) or syntactic biases (♥), compared with jointly training to debias against both bias types. Δ: differences with baseline CE loss.

Bias-only models: We use the hypothesis-only and syntactic bias-only models as in Sections **??** and **??**.

Results: Table **??** shows the results. Models trained to be robust to hypothesis biases (♣) do not generalize to HANS. On the other hand, models trained to be robust on HANS (♥) use a powerful bias-only model resulting in a slight improvement on MNLI mismatched hard dev set. We expect a slight degradation when debiasing for both biases since models need to select samples accommodating both debiasing needs. The jointly debiased models successfully obtain improvements on both datasets, which are close to the improvements on each dataset by the individually debiased models.

5 Transfer Performance

To evaluate how well the baseline and proposed models generalize to solving textual entailment in domains that do not share the same annotation biases as the large NLI training sets, we take trained NLI models and test them on several NLI datasets.

Datasets: We consider a total of 12 different NLI datasets. We use the 11 datasets studied by **?**. These datasets include MNLI, SNLI, SciTail (**?**), AddOneRTE (ADD1) (**?**), Johns Hopkins Ordinal Commonsense Inference (JOCI) (**?**), Multiple Premise Entailment (MPE) (**?**), Sentences Involving Compositional Knowledge (SICK) (**?**), and three datasets from **?** which are automatically generated from existing datasets for other NLP tasks including: Semantic Proto-Roles (SPR) (**?**), Definite Pronoun Resolution (DPR) (**?**), FrameNet Plus (FN+) (**?**), and the GLUE benchmark's diagnostic test (**?**). We additionally consider the Quora Question Pairs (QQP) dataset, where the task is to determine whether two given questions are semantically matching (duplicate) or not. As in **?**, we interpret duplicate question pairs as an entailment relation and neutral otherwise. We use the same split ratio mentioned by **?**.

Since the datasets considered have different label spaces, when evaluating on each target dataset, we map the model's labels to the corresponding target dataset's space. See Appendix **??** for more details.

We strictly refrained from using any out-of-domain data when evaluating on the unbiased split of the same benchmark in Section **??**. However, as shown by prior work (**?**), since different NLI target datasets contain different amounts of the bias found in the large-scale NLI dataset, we need to adjust the amount of debiasing according to each target dataset. We consequently introduce a hyperparameter α for PoE to modulate the strength of the bias-only model in ensembling. We follow prior work (**?**) and perform model selection on the dev set of each target dataset and then report results on the test set.[4] We select hyper-parameters γ, α from $\{0.4,0.6,0.8,2,3,4,5\}$.

Results: Table **??** shows the results of the debiased models and baseline with BERT. As shown in prior work (**?**), the MNLI datasets have very similar biases to SNLI, which the models are trained on, so we do not expect any improvement in the relative performance of our models and the baseline for MNLI and MNLI-M. On all the remaining datasets, our proposed models perform better than the baseline, showing a substantial improvement in generalization by using our debasing techniques. We additionally

[4] Since the test sets are not available for MNLI, we tune on the matched dev set and evaluate on the mismatched dev set or vice versa. For GLUE, we tune on MNLI mismatched dev set.

Data	CE	DFL	Δ	PoE	Δ
SICK	57.05	57.91	+0.9	57.28	+0.2
ADD1	87.34	88.89	+1.5	87.86	+0.5
DPR	49.50	50.68	+1.2	50.14	+0.6
SPR	59.85	61.41	+1.6	62.45	+2.6
FN+	53.16	54.77	+1.6	53.51	+0.4
JOCI	50.06	51.13	+1.1	50.85	+0.8
MPE	69.50	70.2	+0.7	70.1	+0.6
SCITAIL	67.64	69.33	+1.7	71.40	+3.8
GLUE	54.08	54.80	+0.7	54.71	+0.6
QQP	67.78	69.28	+1.5	68.61	+0.8
MNLI	74.40	73.58	-0.8	73.61	-0.8
MNLI-M	73.98	74.0	0.0	73.49	-0.5

Table 6: Accuracy results of models with BERT transferring to new target datasets. All models are trained on SNLI and tested on the target datasets. **Δ** are absolute differences between our methods and the CE loss baseline.

compare with **?** in Appendix **??** and show that our methods substantially surpass their results.

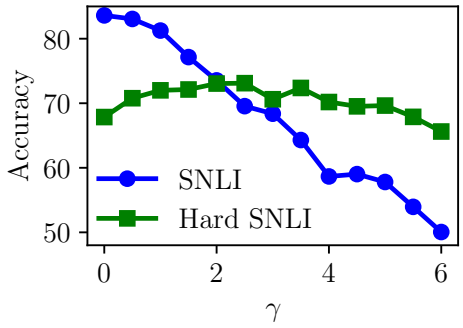

Figure 2: Accuracy of InferSent model trained with DFL, on the SNLI test and SNLI hard sets for different γ.

6 Discussion

Analysis of Debiased Focal Loss: As expected, improving the out-of-domain performance could come at the expense of decreased in-domain performance since the removed biases are useful for performing the in-domain task. This happens especially for DFL, in which there is a trade-off between in-domain and out-of-domain performance that depends on the parameter γ, and when the baseline model is not very powerful like InferSent. To understand the impact of γ in DFL, we train an InferSent model using DFL for different values of γ on the SNLI dataset and evaluate its performance on SNLI test and SNLI hard sets. As illustrated in Figure **??**, increasing γ increases debiasing and thus hurts in-domain accuracy on SNLI, but out-of-domain accuracy on the SNLI hard set is increased within a wide range of values (see a similar plot for BERT in Appendix **??**).

Correlation Analysis: In contrast to **?**, who encourage only the encoder to not capture the unwanted biases, our learning strategies influence the parameters of the full model to reduce the reliance on unwanted patterns more effectively. To test this assumption, in Figure **??**, we report the correlation between the element-wise loss of the debiased models and the loss of a bias-only model on the considered datasets.

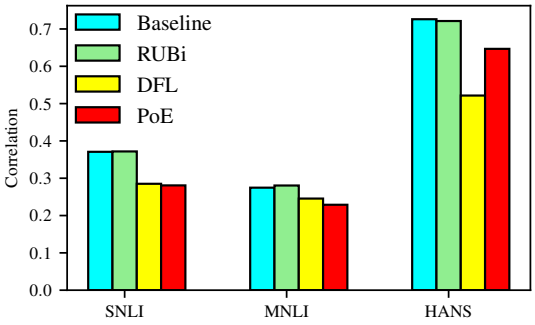

Figure 3: Pearson correlation between the element-wise cross-entropy loss of the debiasing models and the bias-only model trained on each dataset.

The results show that compared to the baselines, our debiasing methods, DFL and PoE, reduce the correlation to the bias-only model, confirming that our models are effective at reducing biases. Interestingly, on MNLI, PoE has less correlation with the bias-only model than DFL and also has better performance on the unbiased split of this dataset. On the other hand, on the HANS dataset, DFL loss is less correlated with the bias-only model than PoE and also obtains higher performance on the HANS dataset.

7 Conclusion

We propose two novel techniques, product-of-experts and debiased focal loss, to reduce biases learned by neural models, which are applicable whenever one can specify the biases in the form of one or more bias-only models. The bias-only models are designed to leverage biases and shortcuts in the datasets. Our debiasing strategies then work by adjusting the cross-entropy loss based on the performance of these bias-only models, to focus learning on the hard examples and down-weight the importance of the biased examples. Additionally, we extend our methods to combat multiple bias patterns simultaneously. Our proposed debiasing

techniques are model agnostic, simple, and highly effective. Extensive experiments show that our methods substantially improve the model robustness to domain-shift, including 9.8 points gain on FEVER symmetric test set, 7.4 on HANS dataset, and 4.8 points on SNLI hard set. Furthermore, we show that our debiasing techniques result in better generalization to other NLI datasets. Future work may include developing debiasing strategies that do not require prior knowledge of bias patterns and can automatically identify them.

Acknowledgments

We would like to thank Daniel Andor and Suraj Srinivas for their helpful comments. We additionally would like to thank the authors of **????** for their support to reproduce their results. This research was supported by the Swiss National Science Foundation under the project Learning Representations of Abstraction for Opinion Summarization (LAOS), grant number "FNS-30216". Y.B. was supported by the Harvard Mind, Brain, and Behavior Initiative.

A Fact Verification

Base model: We fine-tune all models using BERT for 3 epochs and use the default parameters and default learning rate of 2e−5.

Bias-only model: Our bias-only classifier is a shallow nonlinear classifier with 768, 384, 192 hidden units with Tanh nonlinearity.

B Natural Language Inference

Base model: InferSent uses a separate BiLSTM encoder to learn sentence representations for premise and hypothesis. It then combines these embeddings following **?** and feeds them to the default nonlinear classifier. With InferSent we train all models for 20 epochs as default without using early-stopping. We use the default hyper-parameters and following **?**, we set the BiLSTM dimension to 512. We use the default nonlinear classifier with 512 and 512 hidden neurons with Tanh nonlinearity. With BERT, we finetune all models for 3 epochs.

Bias-only model: For debiasing models using BERT, we use the same shallow nonlinear classifier explained in Appendix **??**, and for the ones using InferSent, we use a shallow linear classifier with 512 and 512 hidden units.

Results: Table **??** shows results on the MNLI matched development and hard test sets.

Loss	BERT			InferSent		
	MNLI	Hard	Δ	MNLI	Hard	Δ
Development set results						
CE	84.41	**76.56**		69.97	**57.03**	
RUBi	84.48	77.13	+0.6	70.51	57.97	+0.9
DFL	83.72	77.37	+0.8	60.78	57.88	+0.9
PoE	84.58	**78.02**	**+1.5**	66.02	**59.37**	**+2.3**
Test set results						
None	84.11	75.88		—	—	—
PoE	84.11	**76.81**	**+0.9**	—	—	—

Table 7: Results on the MNLI matched benchmark and MNLI matched hard set. **Δ** are absolute differences with CE loss.

C Syntactic Bias in NLI

Base model: We finetune all models for 3 epochs.

Bias-only model: We use a nonlinear classifier with 6 and 6 hidden units with Tanh nonlinearity.

Results: Table **??** shows the performance for each label (entailment and non_entailment) on individual heuristics of the HANS dataset.

Loss	HANS		
	Constituent	Lexical	Subsequence
gold label: Entailment			
CE	98.98±0.6	96.41±0.8	99.72±0.1
RUBi	99.22±0.3	95.59±0.8	99.50±0.3
DFL	90.90±4.3	84.78±5.0	94.33±4.9
PoE	97.24±1.9	92.16±0.9	98.58±0.5
gold label: Non-entailment			
CE	20.12±5.8	48.86±5.7	7.18±0.7
RUBi	21.89±7.0	46.82±12.5	7.58±2.3
DFL	50.20±9.2	71.06±3.1	24.28±4.4
PoE	36.08±5.1	59.18±8.0	14.63±3.0

Table 8: Accuracy for each label (entailment or non-entailment) on individual heuristics of HANS.

D Transfer Performance

Mapping: We train all models on SNLI and evaluate their performance on other target datasets. SNLI contains three labels, contradiction, neutral, and entailment. Some of the datasets we consider contain only two labels. In the case of labels *entailed* and *not-entailed*, as in DPR, we map contradiction and neutral to the not-entailed class. In the case of labels *entailment* and *neutral*, as in SciTail, we map contradiction to neutral.

Comparison with ?: We modified the implementations of **?** and corrected some implementation issues in the InferSent baseline (**?**). Compared to the original InferSent implementation, the main differences in our implementation include: (a) We incorporated the fixes suggested for the bugs in the implementation of mean/max-pooling over BiLSTM in the InferSent baseline[5] (b). We additionally observed that the aggregation of losses over each batch was computed with the average instead of the intended summation and we corrected it.[6] (c) We followed the implementation of InferSent and we removed out-of-vocabulary (OOV) words from the sentence representation, while **?** keep them by introducing an OOV token. We additionally observed during the pre-processing of some of the

[5] https://github.com/facebookresearch/InferSent/issues/51

[6] The same observation is reported in https://github.com/facebookresearch/InferSent/pull/107.

Data	CE	DFL	Δ%	PoE	Δ%	M1	Δ%	M2	Δ%
SICK	54.09	55.00	1.68	55.79	3.14	49.77	-7.99	49.77	-7.99
ADD1	75.19	78.29	4.12	77.00	2.41	67.44	-10.31	67.44	-10.31
DPR	49.95	50.59	1.28	49.95	0.00	50.87	1.84	50.87	1.84
SPR	41.31	47.95	16.07	50.50	22.25	51.51	24.69	51.51	24.69
FN+	48.65	49.58	1.91	49.35	1.44	53.23	9.41	53.23	9.41
JOCI	46.47	46.48	0.02	47.53	2.28	44.83	-3.53	44.83	-3.53
MPE	60.60	60.70	0.17	61.80	1.98	56.40	-6.93	56.40	-6.93
SCITAIL	64.25	65.19	1.46	63.17	-1.68	56.40	-12.22	56.40	-12.22
GLUE	48.73	46.83	-3.90	49.09	0.74	43.93	-9.85	43.93	-9.85
QQP	61.80	66.24	7.18	66.36	7.38	62.46	1.07	62.46	1.07
MNLI	56.99	56.70	-0.51	56.59	-0.70	51.72	-9.25	51.72	-9.25
MNLI-M	57.01	57.75	1.30	57.84	1.46	53.99	-5.30	53.99	-5.30
Average	—	—	2.57	—	3.39	—	-2.36	—	-2.36

Table 9: Accuracy results of models with InferSent transferring to new target datasets. All models are trained on SNLI and tested on the target datasets. M1 and M2 are our re-implementation of **?**. **Δ** are relative differences in percentage with respect to CE loss.

target datasets in the implementation of **?**, some of the samples are not considered due to the preprocessing issues. We fix the pre-processing issues and evaluate our models and our reimplementations of **?** on the same corpora. We set the BiLSTM dimension to 512 across all models. Note that **?** use BiLSTM dimension of 2048, and due to the mentioned differences in implementations and datasets, the results reported in **?** are not comparable. However, we still on average surpass their reported results substantially. Our reimplementations and scripts to reproduce the results are publicly available in https://github.com/rabeehk/robust-nli-fixed.

As used in prior work to adjust the learning-rate of the bias-only and baseline models (**?**), we introduce a hyperparameter β for the bias-only model to modulate the loss of the bias-only model in ensembling. We sweep hyper-parameters γ, α over $\{0.02, 0.05, 0.1, 0.6, 2.0, 4.0, 5.0\}$ and β over $\{0.05, 0.2, 0.4, 0.8, 1.0\}$. Table **??** shows the results of our debiasing models (DFL, PoE), our re-implementations of proposed methods in **?** (M1, M2), and the baseline with InferSent (CE). The DFL model outperforms the baseline in 10 out of 12 datasets, while the PoE model outperforms the baseline in 9 datasets and does equally well on the DPR dataset. As shown in prior work (**?**), the MNLI dataset has very similar biases to SNLI, which the models are trained on, so we do not expect any improvement in the relative performance of our models and the baseline for MNLI dataset. Interestingly, our methods obtain improvement on MNLI-M, in which the test data differs from training distribution. Our proposed debiasing methods, PoE and DFL, are highly effective, boosting the relative generalization performance of the baseline by 3.39% and 2.57% respectively, significantly surpassing the prior work of **?**. Compared to M1 and M2, our methods outperform them on 9 datasets, while they do better on two datasets of SPR and FN+, and slightly better on the DPR dataset. However, note that DPR is a very small dataset and all models perform close to random-chance on this dataset.

E Analysis of Debiased Focal Loss

Figure **??** shows the impact of γ on BERT trained with DFL.

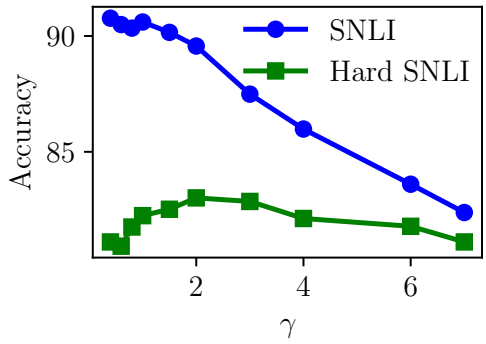

Figure 4: Accuracy of the BERT model trained with DFL, on SNLI and SNLI hard sets for different γ.