A Generative Approach for Mitigating Structural Biases in Natural Language Inference

Anonymous ACL submission

Abstract

Many natural language inference (NLI) datasets contain biases that allow models to perform well by only using a biased subset of the input, without considering the remainder features. For instance, models are able to classify samples by only using the hypothesis, without learning the true relationship between it and the premise. These structural biases lead discriminative models to learn unintended superficial features and generalize poorly out of the training distribution. In this work, we reformulate the NLI task as a generative task, where a model is conditioned on the biased subset of the input and the label and generates the remaining subset of the input. We show that by imposing a uniform prior, we obtain a provably unbiased model. Through synthetic experiments, we find that this approach is highly robust to large amounts of bias. We then demonstrate empirically on two types of natural bias that this approach leads to fully unbiased models in practice. However, we find that generative models are difficult to train and generally perform worse than discriminative baselines. We highlight the difficulty of the generative modeling task in the context of NLI as a cause for this worse performance. Finally, by fine-tuning the generative model with a discriminative objective, we reduce the performance gap between the generative model and the discriminative baseline, while allowing for a small amount of bias.

1 Introduction

Natural language processing (NLP) datasets are plagued with artifacts and biases, which allow models to perform tasks without learning the desired underlying language capabilities. For instance, in natural language inference (NLI) datasets, models can predict an entailment relationship y from the hypothesis text H alone, without considering the premise P at all (Gururangan et al., 2018; Poliak et al., 2018). Another identified source of bias is lexical overlap between P and H, which is associated with an entailment prediction (McCoy et al., 2019). We refer to such biases as *structural biases*, cases where an undesired subset of the input alone incorrectly identifies the label. Relying on such biases results in poor out-of-distribution (o.o.d) generalization when models are applied to data without bias. Furthermore, models that contain such biases may make surprising predictions when the bias is present, causing problems in critical systems.

A line of work has attempted to improve the performance on o.o.d datasets by proposing different objective functions (e.g., Utama et al., 2020a; Karimi Mahabadi et al., 2020). However, these methods typically still result in a significant gap between the performance in and out of distribution, which indicates that the models are still biased. Table 1 shows this gap, which we term the o.o.d generalization gap (Δ).

In this work, we reformulate classification as a generative task, where the model's task is to generate the remainder features R conditioned on the biased features B and the label y. Using Bayes' Rule, we decompose the posterior $p(y \mid B, R)$ into the likelihood $p(R \mid y, B)$ and the prior $p(y \mid B)$. This reformulation lets us control the amount of bias present in the final model. By setting a uniform prior we can obtain a provably unbiased model. We denote this generative model as **GEN.**.

To assess the extent to which a given model is biased w.r.t a specific structural bias, we consider two metrics: the o.o.d generalization gap and the correlation between a model and a biased model $p(y \mid B)$, such as a hypothesis-only or overlap-only model. We first experiment with injecting synthetic bias into a fraction of the training set and evaluating on test sets with and without that bias. We find that the discriminative model's performance decreases as the amount of bias increases, while GEN maintains similar performance at all bias levels. Moreover, the biased-ness of the discriminative model increases, while GEN remains unbiased.

	SNLI			MNLI		
	Test	Hard test	Δ	Test	Hard test	Δ
Utama et al. (2020a)	–	–	–	82.8	79.8	+3.00
Karimi Mahabadi et al. (2020)	**89.57**	**83.01**	+6.56	**83.47**	76.83	+6.64
Sanh et al. (2021)	–	–	–	83.32	**77.63**	+5.69
Gururangan et al. (2018)	86.5	72.7	+13.8	76.5	64.4	+11.1
Stacey et al. (2020)	79.39	69.92	+9.47	–	–	–
GEN (BERT)	65.53	66.18	**−0.65**	58.55	57.33	**+1.22**
GEN (BART)	70.58	72.19	−1.61	64.09	65.74	−1.65

Table 1: Results on regular and hard (o.o.d) test sets of SNLI and MNLI. Prior work exhibits large o.o.d generalization gaps (Δ), while our generative approach reduces the gap significantly.

Next, we experiment with two kinds of natural bias: hypothesis-only and overlap. We demonstrate that GEN is unbiased compared to the discriminative baseline as measured by its low Δ and low absolute correlation with a biased model (ρ).

However, while our approach leads to unbiased models, it performs worse than the discriminative baseline even on o.o.d data. We then identify and quantify several causes for the poor performance of GEN. We show that generative modeling is a more challenging task than discriminative modeling, and that it requires learning a large amount of spurious signal compared to the discriminative model.

Finally, to mitigate the difficulty of the generative modeling task, we fine-tune GEN with a discriminative objective (Lewis and Fan, 2019). While this leaks some bias into the model, the final model (denoted as **GEN-FT**) matches or surpasses the discriminative baseline while maintaining a relatively small o.o.d generalization gap.

To conclude, our contributions are as follows:

- We develop a generative modeling approach, which provably eliminates structural biases in natural language understanding tasks.
- We demonstrate experimentally on two bias types and different NLI datasets that this approach leads to unbiased models.
- We analyze the strengths and weaknesses of the generative model.
- We show how discriminative fine-tuning improves the generative model, while allowing some bias to leak into the model.

2 Related Work

2.1 Biases and Artifacts

Many natural language understanding (NLU) datasets contain biases or artifacts, superficial features that are associated with a certain label. Examples include hypothesis-only biases in NLI such as negation words in the hypothesis being correlated with a contradiction label (Poliak et al., 2018; Gururangan et al., 2018). Similar one-sided biases have been found in other tasks, including visual question answering (VQA) (Agrawal et al., 2018; Das et al., 2019), reading comprehension (Kaushik and Lipton, 2018), and fact verification (Schuster et al., 2019). Another kind of bias identified in NLI is lexical overlap, which is correlated with an entailment decision in NLI datasets (McCoy et al., 2019). We view all these cases as structural biases, cases where the input can be split into two disjoint sets, of the biased features and the remainder features.

The existence of structural biases in datasets allows models to perform unreasonably well when given access only to the biased features, such as a hypothesis-only model being able to predict entailment without access to the premise. The bias learned by the model manifests in poor o.o.d generalization when evaluated on a test set where the training set correlation between the biased features and a certain label does not hold.

2.2 Mitigation Strategies

Common approaches for improving o.o.d generalization combine the main model with a bias model, such as a hypothesis-only model. For instance, a bias model may be trained adversarially, making the main model perform worse when the bias model performs well (Belinkov et al., 2019b; Stacey et al., 2020). Others use a bias model to modulate the main model's predictions in different ways (He et al., 2019; Karimi Mahabadi et al., 2020; Utama et al., 2020b). All these approaches use discriminative models to estimate $p(y \mid P, H)$. Moreover, they typically still result in a gap between in- and out-of-distribution performance.

In contrast, we propose a novel generative formulation of the NLI task, which leads to an unbiased model, in theory, and in practice. Belinkov et al. (2019a) also proposed to solve a generative problem, modeling $p(P \mid y, H)$, in order to encourage the model to consider the premise in its predictions. However, they ended up not using a generative model; rather, they approximated it with discriminative models. Lewis and Fan (2019) used a generative model for a different task, VQA, and found it improves generalization from biased training data. While our basic approach is similar, we analyze the generative model more rigorously, investigate the effect of different modeling options, and focus on quantifying the model's bias.

3 Structural Bias

Consider the general case of a classification task, for which we wish to build a model $p_\theta(y|X)$ where y is a low-dimensional label and X is an arbitrarily large set of features. The model is trained on an empirical training set $\mathcal{D} = \{(X_i, y_i)\}_{i=1}^N$. The dataset is constructed by humans, and inadvertently contains *structural biases*. We define a structural bias as a case where, if the input X is split into two disjoint sets X=$(B, R = X - B)$, the label y can be learned to be reliably predicted given only B. For most choices of B this is not a problem, but in some cases, the subset represents an externally imposed constraint that needs to be maintained or an externally imposed understanding of how the model should operate.

This formulation comprises a broad set of commonly considered biases. For example, in the NLI task, $X = (P, H)$ where P and H are the premise and hypothesis. If we choose the split $B = H$, we arrive at the hypothesis-only bias. This is an undesirable bias because as humans we know that NLI is impossible if one is only given the hypothesis.

Taking different splits leads to representations of different biases. For instance, we can model the lexical overlap bias under the structural bias framework with the subset $B = P \bigcap H$. NLI models should perform no better than chance when given only the overlapping tokens between P and H.

Finally, this formulation extends beyond NLI and NLP to broader biases. For example, if one of the features in X is a protected characteristic s (e.g., gender or race), $B = s$. Then, depending on the task, an undesirable structural bias may exist if a model can learn to predict y given s.

We denote these biases as structural biases because they are defined through the structure $B \subset X$, rather than specific known patterns in the data. For example, in the hypothesis-only case, this formulation does not require knowledge about what aspects of the hypothesis allow a hypothesis-only model to predict the label (e.g., negation words), only that somehow the hypothesis alone incorrectly gives a signal about the label. Thus, this type of bias is broader than specific known biases such as the presence of negation words, but narrower than unknown biases because it requires some knowledge of where the bias might be found.

3.1 Generative Classifiers Eliminate Structural Bias

Generative classifiers are models that make predictions according to Bayes' Rule. The generative classifier framework provides a principled way of handling structural bias:

$$\begin{aligned} p_\theta(y \mid X) &= p_\theta(y \mid B, R) \qquad (1) \\ &= \frac{p_\theta(R \mid y, B) p_\theta(y \mid B)}{p_\theta(R \mid B)} \\ &= \frac{p_\theta(R \mid y, B) p_\theta(y \mid B)}{\sum_{y'} p_\theta(R \mid y', B) p_\theta(y' \mid B)}. \end{aligned}$$

We emphasize that under this framework, one may separately model $p_\theta(R \mid y, B)$ and $p_\theta(y \mid B)$, but the marginal likelihood must be constructed by marginalizing over the product of those components rather than estimated separately.

Separating the bias component gives explicit control over a given structural bias in the model. Formally, consider the ability of any model to predict the label given the bias subset, $p(y \mid B)$, defined by marginalizing out the remainder features:

$$p(y \mid B) = \int p_\theta(y \mid R, B) p_\theta(R \mid B) \mathrm{d}R. \qquad (2)$$

For a discriminative model this may take any value, but for a generative classifier this becomes:

$$\begin{aligned} p(y \mid B) &= \int p_\theta(y \mid R, B) p_\theta(R \mid B) \mathrm{d}R \qquad (3) \\ &= \int p_\theta(R \mid y, B) p_\theta(y \mid B) \mathrm{d}R \\ &= p_\theta(y \mid B) \int p_\theta(R \mid y, B) \mathrm{d}R = p_\theta(y \mid B). \end{aligned}$$

Therefore, for any given structural bias, the ability of the model to rely on the bias alone, $p(y \mid B)$, can be eliminated in a principled way by training a generative model to learn $p_\theta(R \mid y, B)$ and setting

$p_\theta(y \mid B) = \text{Uniform}(\mathcal{Y})$. R and B are collections of tokens, so the actual training process amounts to training a standard encoder–decoder model. Predictions are made using Eq. 1 at inference time. Unlike other methods, this approach does not require a specific model for $p_\theta(y \mid B)$; it simply requires the *desired* $p_\theta(y \mid B)$, which is often uniform.

3.2 Measuring Structural Bias

Typically, debiasing methods are evaluated by measuring the accuracy of the resulting model on a "hard" test set, a subset of the test set for which a bias-only model $p(y \mid B)$ predicts the incorrect label. While this captures overall quality, it does not assess the extent to which bias remains. For some applications, the overall quality on non-biased data is a reasonable final objective, but for other applications complete removal of bias is critical.

To quantify the remaining biased-ness of a given model, we consider two metrics: the difference between the accuracy of the model on the standard test set and its accuracy on a "hard" set created with respect to the bias in question, which we term the o.o.d generalization gap (Δ), and the correlation (ρ) between the predictions of a given model and a fully biased model, i.e., $p(y \mid B)$.

A truly unbiased model will give a similar performance on the original test set and the hard test set, because it cannot rely on the predictive power of B in the original test set even when it is present. Thus low values of Δ indicate the model is unbiased.

Similarly, a model that consistently makes similar decisions to the fully biased model $p(y \mid B)$ in the original test set is likely using only the biased features B as the fully biased model. Therefore, a larger ρ gives additional evidence that a specific structural bias remains in a given model.

4 Experiments

In all experiments, we estimate $p(R \mid y, B)$ with an encoder-decoder model, with inputs (y, B) and output R. To condition on y, we prefix a label-specific token to B. We then train the model as a conditional generative model, by fine-tuning BERT (Devlin et al., 2019) or BART (Lewis et al., 2020) with the standard auto-regressive cross-entropy loss. At test time, we attach all possible label tokens to each B and pick $\hat{y} = \arg\max_{y \in \mathcal{Y}} p_\theta(R|y, B)$.

4.1 Synthetic Experiment

To empirically verify the analysis in Section 3, we construct a synthetic experiment by artificially injecting a hypothesis-only bias into an NLI dataset, similarly to He et al. (2019). We use MNLI (Williams et al., 2018), an English NLI dataset, as the base NLI dataset. For each example, we add one of three tokens to the beginning of the hypothesis, each token corresponding to a label. With probability p the token corresponds to the true label and with probability $1 - p$ the token is randomly selected from the three labels. The result is that p directly controls the amount of hypothesis-only bias present in the data.

Figure 1 shows the results when training with synthetic bias in MNLI, for different values of p, and evaluating on MNLI dev hard (without synthetic bias), a subset that a hypothesis-only model predicts incorrectly. The discriminative model's performance degrades gradually as p increases, while GEN maintains similar performance. At high levels of p, the discriminative model falls below the generative one, indicating that the presence of large amounts of bias precludes the discriminative model from learning the task effectively. Figure 2 shows the two biased-ness metrics, calculated for the gen-

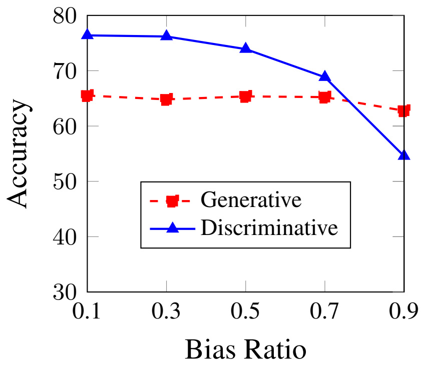

Figure 1: Results for models trained with synthetic bias and evaluated on MNLI dev hard without bias.

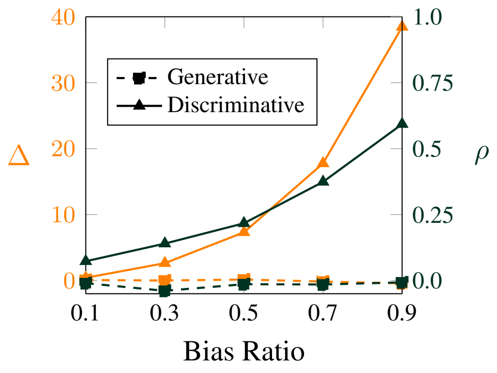

Figure 2: The o.o.d generalization gap (Δ) and the correlation to a bias model (ρ) of generative and discriminative models. ρ is calculated on an unbiased test set. Appendix A.2 shows correlations on a biased test set.

erative and discriminative models across a range of p values. For each p, Δ is calculated from the difference in accuracy for a given model between a version of the dev set with the synthetic bias included as in training, and a version of the dev set with the synthetic bias token randomly chosen for each example. The fully biased model used as the reference when calculating ρ is a model that always selects the label that corresponds with the synthetic bias token prefixed to the hypothesis. According to both metrics, as the bias ratio p increases, the discriminative model quickly becomes significantly biased while GEN remains entirely unbiased.

4.2 Hypothesis-only Bias

We train our models on the (English) Stanford Natural Language Inference dataset (SNLI; Bowman et al. 2015) and on the MNLI dataset, two NLI datasets that are known to contain hypothesis-only biases (Poliak et al., 2018; Gururangan et al., 2018; Tsuchiya, 2018). We evaluate models on the available in-distribution test sets and on o.o.d test sets that have fewer or no hypothesis-only biases. For SNLI, we use the hard set provided by Gururangan et al. (2018). For MNLI, we use the blind evaluation test and hard test sets for MNLI matched.

We experiment with two kinds of pre-trained models: a BERT model, which is combined in an encoder-decoder model during the fine-tuning stage, and BART, a pre-trained encoder-decoder, which is then fine-tuned. For the first kind of model, we used BERT as both an encoder and a decoder. The encoder is a regular BERT model, and the decoder is a BERT model with a language modeling layer, which starts generating from the "CLS" token. All models are taken from the Transformers library (Wolf et al., 2019). Both models are fine-tuned with either the baseline discriminative objective or our proposed generative formulation.

4.3 Overlap Bias

Another type of bias that has been demonstrated in the MNLI dataset is lexical overlap bias. McCoy et al. (2019) demonstrate that, while somewhat uncommon, lexical overlap, subsequence overlap, and constituent overlap between the premise and hypothesis give a strong signal for entailment. Like hypothesis-only bias, this signal comes from peculiarities of the dataset creation process. For a model performing actual NLI, the overlap of words between the hypothesis and the premise should not give any indication of the label. This is emphasized by McCoy et al. (2019), as they create a separate label-balanced evaluation set where each example has a high overlap.

To treat overlap bias in the generative formulation, we set $B = P \bigcap H$. Specifically, we concatenate the premise and hypothesis and mask out any tokens that do not appear in both of them. The input to the encoder of GEN is then the label y followed by this partially masked concatenation. For simplicity, the output of GEN is the unmasked concatenation of P and H. In principle, we do not need to output the unmasked tokens, but this simplifies training and remains probabilistically valid.[1]

Because this setup is closely connected to the way the BART model is pretrained, we experiment solely with the BART model for this configuration.

While not traditionally studied in the overlap bias case, we perform the same analysis as in the hypothesis-only bias by constructing a hard set for overlap bias. We train a discriminative model that predicts the label from the masked concatenated premise–hypothesis input, and filter the MNLI dev set for examples where this model is incorrect.[2]

5 Results

5.1 GEN Reduces the Generalization Gap

Hypothesis-only bias Table 2 shows the results of the proposed generative model and the discriminative baseline in the case of hypothesis-only bias. For GEN, we show results with either a hypothesis-only prior for $p(y \mid H)$ or a uniform prior. The generative approach with the uniform prior leads to nearly identical accuracy on the i.i.d and o.o.d test sets, that is, unbiased models as measured by low o.o.d generalization gap (Δ between -2 and 3). In contrast, the discriminative model has much larger gaps (Δ of at least 9 on SNLI and 7 on MNLI), meaning that it is a more biased model. GEN with a hypothesis-only prior also exhibits large generalization gaps, demonstrating the bias leak in this model. Obviously, a hypothesis-only model is the most biased, with the largest gaps.

These results also show the advantage of using a pre-trained encoder-decoder (BART) compared to plugging a pre-trained encoder (BERT) and fine-tuning it as an encoder-decoder. While both generative models are unbiased, BART is more amenable to the generative fine-tuning than BERT, with overall better results. For this reason, we only report results with BART henceforth.

[1]An example for the data preparation is in Appendix A.4.

[2]As we cannot use the hidden test to filter based on labels, we use dev matched/mismatched for val./eval. respectively.

	Model	SNLI Test	SNLI Hard test	SNLI Δ	MNLI Test	MNLI Hard test	MNLI Δ
BERT	Bias-only	70.82±0.6	32.09±1.7	38.73±1.2	59.77±0.4	34.41±2.3	25.36±2.1
	Discriminative	90.49±0.2	80.55±0.3	9.94±0.1	84.08±0.4	76.27±0.3	7.81±0.2
	GEN, hyp-only	81.42±0.5	61.39±1.4	20.02±0.9	68.5±0.3	52.24±1.4	16.21±1.5
	GEN, uniform	65.86±0.3	66.74±0.5	**−0.88**±0.3	56.98±0.7	54.73±0.2	2.26±0.5
BART	Hypothesis-only	70.37±0.3	31.61±0.3	38.76±0.1	57.89±2.3	37.83±1.6	20.05±3.9
	Discriminative	**90.78**±0.3	**81.04**±0.6	9.74±0.3	**85.67**±0.1	**78.84**±0.4	6.83±0.4
	GEN, hyp-only	84.36±0.1	67.22±0.8	17.14±0.7	73.85±0.6	60.79±0.6	13.06±0.2
	GEN, uniform	70.80±0.2	73.16±0.9	−2.36±0.7	64.22±0.4	64.11±1.0	**0.11**±0.8

Table 2: Comparison between discriminative baselines and generative models, with Hyp-only or uniform prior, in the hypothesis-only bias case.

Model	Dev	Hard dev	Δ
Bias-only	56.32±0.3	9.37±8.3	46.95±8.5
Disc.	**86.44**±0.5	**79.72**±0.8	6.73±0.2
GEN	63.67±1.1	65.56±0.6	**−1.88**±0.4

Table 3: Comparison of discriminative and generative models (fine-tuned from BART) in the lexical overlap bias case. GEN was trained with a uniform prior.

Model	Hyp-SNLI	Hyp-MNLI	Overlap
Disc.	0.271	0.223	0.171
GEN	−0.025	−0.009	−0.043
Majority	0.005	0.055	0.016
Uniform	−0.018	−0.006	0.007

Table 4: Correlations of discriminative, generative, majority, and uniform models with bias models, on hyp-only (on SNLI/MNLI) and overlap bias (on MNLI).

Overlap bias Table 3 shows similar results in the case of overlap bias on a hard set w.r.t this bias.[3] GEN exhibits a lower generalization gap (Δ) than the discriminative baseline. As expected, the overlap bias model shows the greatest gap.

While the generative approach leads to unbiased models for both bias types, it also performs significantly worse than the discriminative model, on both in-distribution and o.o.d test sets. We return to this issue in Sections 6 and 7.

5.2 GEN is Uncorrelated with a Bias Model

Table 4 shows correlations ρ of GEN and the discriminative baseline with a bias-only model. In the hypothesis-only case, the models were trained on SNLI or MNLI and correlations were measured on predictions on SNLI test or MNLI dev mismatched, respectively. In the overlap case, the models were trained on MNLI and correlations were measured on MNLI dev mismatched.

In both bias types, the discriminative model predictions are much more correlated with the bias models than the predictions of the generative models. In fact, the correlations of the generative models are as low as those of a majority model or a uniform model, which is unbiased by construction.

[3]For results on HANS evaluation sets, see Appendix A.1.

6 Evaluating Generated Premises

So far, we have only used GEN to score existing examples (with teacher forcing), conditioned on the label and the biased features. In this section, we evaluate the quality of its generations when decoding without constraints. For the experiments here, we consider the hypothesis-only bias and evaluate the quality of GEN in generating premises. We use a BART model trained on SNLI and generate premises for all hypotheses in the test set.

To evaluate how well our model can generate premises, we used two metrics: BLEU (Papineni et al., 2002) of the generated premises w.r.t gold premises, to measure the generation quality (higher is better), and self-BLEU (Zhu et al., 2018) to measure the diversity of the generations (lower is more diverse). We report a BLEU value of 0.1078, indicating that the model is not very good at generating premises. We report self-BLEU of 0.8032 for the generated premises compared to 0.5875 for the original premises, suggesting that the generated premises are less diverse. Table 5 also shows examples where, given different hypotheses, the model generates very similar premises.

A possible explanation for the difficulty of the generative task may be found in the nature of NLI

Label	Hypothesis	Generated premise
contradiction	the woman is young	an elderly woman is sitting on a bench with her legs crossed and [...]
entailment		a young woman in a black shirt and jeans is walking down the street
neutral		a woman in a red shirt is sitting on a bench with a bag in her lap

Table 5: Generated premises by GEN from <y, H> pairs. The original premise for the hypothesis was *"A woman with a green headscarf, blue shirt and a very big grin"* and the gold label was "neutral".

	SNLI				MNLI			
Model	**Test**	**Hard test**	Δ	ρ	**Test**	**Hard test**	Δ	ρ
Disc.	**90.78**±0.3	81.04±0.6	9.74±0.3	0.27	**85.67**±0.1	**78.84**±0.4	6.83±0.4	0.22
GEN-FT	86.30±0.4	**82.20**±0.3	**4.09**±0.1	**0.09**	79.66±1.5	76.45±0.7	**3.21**±1.3	**0.07**

Table 6: Fine-tuned model results for hypothesis-only bias. Disc. is the discriminative baseline.

examples in common datasets. In many cases, the relationship is determined by a small number of words in the premise and hypothesis pair. To quantify this, we measured the number of words highlighted as explanations in the e-SNLI dataset (Camburu et al., 2018) and found that less than 21% of words in the premise are highlighted on average.[4] This pattern is reflected also in decisions made by NLI models. By applying gradient attributions,[5] we found that more than 70% of the premise words have low attributions values (between -0.1 to 0.1), with fewer than 6% of the words having absolute values greater than 0.3. This shows that only a small number of words had any significant effect on the model predictions. Table 13 in the appendix shows a qualitative example of this behavior. Finally, this pattern is also reflected in the generations produced by GEN, as demonstrated in Table 5.

7 Discriminative Fine-tuning

The analysis in Section 6 suggests that the central limitation of GEN is that at training time there is no indication of the model's downstream use as a classifier. The model is rewarded at training time for devoting significant capacity to modeling the full high-dimensional distribution of R, even when large parts of that distribution are unimportant for making downstream predictions.

To help GEN in such cases, we experiment with an additional fine-tuning step in which we directly optimize for predictive performance. Specifically, for the fine-tuning step we construct the discriminative distribution using Bayes' Rule in Eq. 1 and use it at *training time* by minimizing the label cross-entropy loss:

$$\mathcal{L}_{ft} = -\sum_{i=1}^{N} \log p_\theta(y_i \mid B_i, R_i) \tag{4}$$

$$= -\sum_{i=1}^{N} \log \frac{p_\theta(R_i \mid y_i, B_i) p_\theta(y_i \mid B_i)}{\sum_{y'} p_\theta(R_i \mid y', B_i) p_\theta(y' \mid B_i)}.$$

Using this objective requires a choice of $p_\theta(y \mid B)$. We explore the impact of different choices for this distribution in App. A.3, but found that using a pretrained and frozen $p_\theta(y \mid B)$ during the fine-tuning step works best. We hypothesize that this setup allows the generative component $p_\theta(R \mid y, B)$ to ignore as much bias as possible. At inference, as we would like to ignore the bias, we take the fine-tuned generative component $p_\theta(R \mid y, B)$ and perform inference the same way as before, using Bayes' Rule with a uniform prior.

The adjusted training procedure is composed of the following steps: 1) Train a discriminative prior model, $p_\theta(y \mid B)$, freeze the weights. 2) Train a generative model, $p_\theta(R \mid y, B)$, as in Section 4. 3) Fine-tune the model using Eq. 4, using the pretrained $p_\theta(y \mid B)$. 4) Test the model using Eq. 1 with a uniform prior.

7.1 Results

Tables 6 and 7 show the results of the fine-tuning pipeline. The fine-tuned generative models (denoted as **GEN-FT**) achieve smaller o.o.d generalization gaps (Δ) and correlations to the biased models (ρ) than the discriminative baselines. GEN-FT is also significantly better than GEN in terms of o.o.d performance, at the expense of slight bias leakage (higher ρ compared to GEN in Table 4). In the case

[4] Of the premises that were highlighted at all.

[5] We computed attributions with Integrated Gradients (Sundararajan et al., 2017) using Captum (Kokhlikyan et al., 2020).

Model	Dev	Hard dev	Δ	ρ
Disc.	**86.44**	**79.72**	6.73	0.171
GEN-FT	79.87	74.98	**4.89**	**0.106**

Table 7: Fine-tuned model results for overlap bias on MNLI mismatched dev set.

of hypothesis-only bias, GEN-FT match or surpass the results of the discriminative baselines on the o.o.d sets. In the overlap bias case, GEN-FT does not match the discriminative model on the o.o.d set, but it narrows the gap.

The above results were obtained using a bias model prior in the fine-tuning step and a uniform prior at inference time. This was the strategy that achieved the lowest generalization gap (Δ) on the dev set while outperforming the discriminative baseline. See Appendix A.3 for an ablation study of additional options.

8 Scalability

Given that the generative approach consumes more compute than the discriminative baseline, it is natural to ask whether it can scale to larger models. To answer this, we experimented with the T5 model (Raffel et al., 2020), an encoder-decoder available in five sizes, from 60M to 11B parameters. We focus on the hypothesis-only bias case in SNLI. We train the generative and discriminative models using regular fine-tuning (also called *model-tuning*), and also experiment with *prompt-tuning* (Lester et al., 2021), a faster and cheaper approach, which adds a small number of learnable tokens to the start of the input, and trains them end-to-end, while the model's original weights stay frozen (Memory and training statistics are found in Table 14, App. B.1).

Figure 3 shows that both the generative and discriminative approaches scale with model size. Prompt-tuning is effective, matching model-tuning performance at larger sizes. In larger models, the generative approach narrows the gap from the discriminative one, but cannot close it. Table 8 shows that with the largest 11B model, the generative approach leads to unbiased models. The table also shows that discriminative fine-tuning is possible at this scale and obtains a similar performance to the discriminative model on the hard set. Prompt-tuning also allows us to hold only one model for the discriminative fine-tuning phase (compared to two models in model-tuning). We conclude that the generative approach is scalable and can be used with very large models to mitigate structural biases.

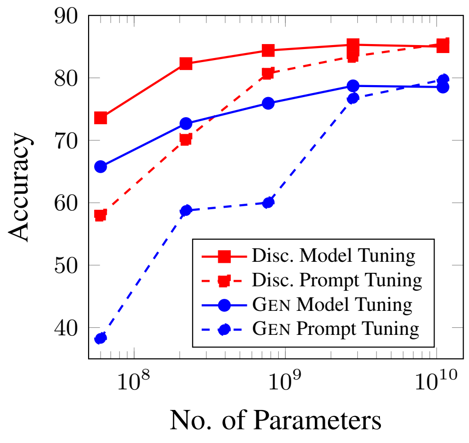

Figure 3: SNLI Hard results with different T5 models.

Method	Model	Test	Hard test	Δ
Model-tuning	Disc.	92.80	85.00	7.8
	GEN	76.76	78.53	−1.77
Prompt-tuning	Disc.	92.88	85.46	7.42
	GEN	79.91	79.70	0.21
	GEN-FT	89.68	85.22	4.46

Table 8: Results on SNLI with T5-XXL (11B) model.

9 Conclusion

Structural biases are common in various NLI datasets and are a major obstacle when trying to create robust systems for this task. We proposed a generative approach for NLI, which leads to unbiased models. We demonstrated that our generative models are robust to large amounts of bias and perform equally well in and out of distribution. This comes, however, with a trade-off, where the generative models perform worse than discriminative baselines. We investigated reasons for the difficulty of training generative NLI models, highlighting the large output space of generating sentences, as opposed to identifying a small subset of words that are often sufficient for solving the task. We showed how to mitigate this problem by fine-tuning GEN with a discriminative objective. Finally, we demonstrated that the method scales efficiently to large language models. Our work lays down a novel formulation for the NLI task, which may be applied to many other natural language understanding tasks. Future work can examine other kinds of bias and different tasks. For instance, if the bias variables are constructed according to protected attributes like race or gender, our approach leads to unbiased models w.r.t the protected attributes.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. 2018. Don't just assume; look and answer: Overcoming priors for visual question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4971–4980. IEEE Computer Society.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019a. Don't take the premise for granted: Mitigating artifacts in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 877–891, Florence, Italy. Association for Computational Linguistics.

Yonatan Belinkov, Adam Poliak, Stuart Shieber, Benjamin Van Durme, and Alexander Rush. 2019b. On adversarial removal of hypothesis-only bias in natural language inference. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 256–262, Minneapolis, Minnesota. 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 *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 9560–9572.

Anubrata Das, Samreen Anjum, and Danna Gurari. 2019. Dataset bias: A case study for visual question answering. *Proceedings of the Association for Information Science and Technology*, 56(1):58–67.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

He He, Sheng Zha, and Haohan Wang. 2019. Unlearn dataset bias in natural language inference by fitting the residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142, Hong Kong, China. Association for Computational Linguistics.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8706–8716, Online. Association for Computational Linguistics.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, et al. 2020. Captum: A unified and generic model interpretability library for pytorch. *arXiv preprint arXiv:2009.07896*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Mike Lewis and Angela Fan. 2019. Generative question answering: Learning to answer the whole question. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. 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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

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

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2021. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*.

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics.

Joe Stacey, Pasquale Minervini, Haim Dubossarsky, Sebastian Riedel, and Tim Rocktäschel. 2020. Avoiding the Hypothesis-Only Bias in Natural Language Inference via Ensemble Adversarial Training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8281–8291, Online. Association for Computational Linguistics.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR.

Masatoshi Tsuchiya. 2018. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020a. Mind the trade-off: Debiasing NLU models without degrading the in-distribution performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8717–8729, Online. Association for Computational Linguistics.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020b. Towards debiasing NLU models from unknown biases. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7597–7610, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, pages arXiv–1910.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, pages 1097–1100. ACM.

A Appendix

A.1 HANS evaluation sets

Given that we study overlap bias, it is natural to evaluate the generative classifier approach on the HANS evaluation set (McCoy et al., 2019). The three HANS evaluation sets are constructed to be label balanced, while containing exclusively examples of significant amount of overlap. Three specific types of overlap are considered: lexical overlap, subsequence overlap, and constituent overlap. See McCoy et al. (2019) for more details.

Table 9 shows the accuracies of the generative classifier and previous results from the literature, reported by Utama et al. (2020a). In general, the accuracies for the generative classifier are low. We hypothesize that this is due to the fact that the examples in the HANS evaluation set are significantly out of distribution compared to the training set, w.r.t the amount of overlap between premise and hypothesis. In the training set, sentences often have 20 or 30 tokens with only 1 or 2 token overlaps. In the HANS set, sentences are shorter and all but 1 or 2 tokens overlap. This makes the input significantly more out of domain for the generative classifier only, which is used to seeing many mask tokens in the input and in the HANS set sees almost no mask tokens.

Model	Lex.	Subseq.	Const.
Hypothesis-only	48.2	48.7	50.4
Discriminative	80.7	55.5	66.3
Learned-mixin	77.5	54.1	63.2
PoE	72.9	65.3	69.6
Conf. reg.	73.3	66.5	67.2
Generative	50.7	57.7	53.2

Table 9: Discriminative and generative models evaluated on the three HANS evaluation sets.

A.2 Correlations

Figure 4 shows the correlations of generative and discriminative models to a bias model under different bias ratios in the synthetic bias case. Here the correlations are calculated on a biased test set, while in Section 5.2 they were calculated on an unbiased test set. The pattern is the same: the discriminative model is become more biased (higher ρ) as the bias ratio increases, while GEN remains unbiased (small ρ).

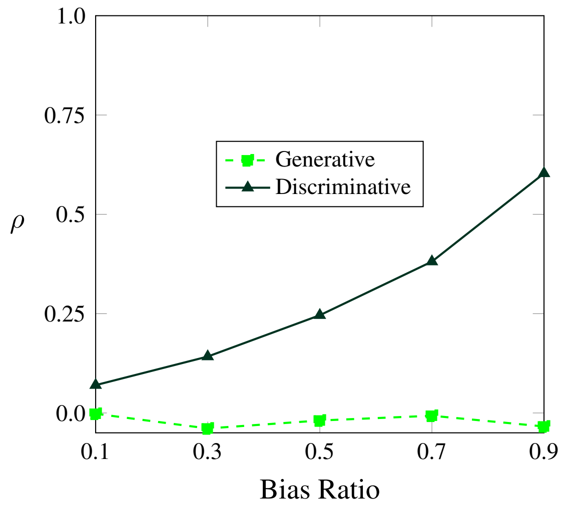

Figure 4: The correlation to a bias model (ρ) of generative and discriminative models under different bias ratios. ρ is calculated on a biased test set, so that each model used the same bias ratio at training and inference time.

A.3 Ablation study of fine-tuning pipeline

Our fine-tuning pipeline allows different ways to combine the steps, such as choosing a prior or whether to use another step of fine-tuning. Table 10 presents an ablation study of the different possible combinations, using BART on SNLI with hypothesis-only bias. (The table shows means and standard deviations of 3 runs with different random seeds.) Row 1 shows the results of GEN, without any fine-tuning; the same model from Section 3. Fine-tuning with a hypothesis-only prior leads to a smaller gap than fine-tuning with a uniform prior (compare rows 3 and 5). We can explain these apparently surprising results by the hypothesis-only prior capturing some of the bias, such that removing it during inference allows the predictions to be less biased. Fine-tuning with a uniform prior does not allow such a decomposition, resulting in a large gap (row 3). In contrast, using a hypothesis-only prior at inference leads to biased predictions (large generalization gaps; rows 2, 4 and 6). These models perform well on the test set (relative to using uniform prior at inference; rows 3, 5, 7), but relatively poorly on the o.o.d set. In fact, maintaining the same kind of prior throughout the pipeline (rows 3 and 6) leads to results similar to the discriminative baseline (row 11).

The fine-tuning step allows a balancing of bias and performance. Fine-tuning with a hypothesis-only prior and using a uniform prior at test time

	Training	Prior: Fine-tuning	Prior: Inference	Dev	Hard Dev	Δ
1	GEN	–	Uniform	71.14±0.4	72.68±0.2	**−1.54**±0.3
2	GEN	–	Hypothesis-only	84.99±0.3	64.29±0.7	20.69±0.9
3	GEN	Uniform	Uniform	**90.32**±0.1	77.17±0.8	13.15±0.7
4	GEN	Uniform	Hypothesis-only	89.31±0.5	70.33±1.9	18.98±1.4
5	GEN	Hypothesis-only	Uniform	87.12±0.5	**80.55**±0.1	6.57±0.5*
6	GEN	Hypothesis-only	Hypothesis-only	90.06±0.0	75.53±0.8	14.53±0.8
7	–	Uniform	Uniform	90.05±0.1	76.54±0.4	13.51±0.3
8	–	Uniform	Hypothesis-only	89.66±0.4	71.71±1.5	17.95±1.2
9	–	Hypothesis-only	Uniform	87.11±0.9	80.53±1.2	6.58±0.6
10	–	Hypothesis-only	Hypothesis-only	90.04±0.2	76.13±0.6	13.91±0.4
11	Discriminative baseline			91.49±0.0	79.59±0.5	11.90±0.5

Table 10: Ablations on SNLI validation set (Dev) with BART-base. Hard Dev was created similarly to SNLI hard (Gururangan et al., 2018). Fine-tuning is done with a discriminative objective, while inference is always using the generative objective. Uniform/Hypothesis-only refers to the kind of prior that was used during this phase. "*" marks the model with the smallest o.o.d generalization gap (Δ) that is better than the discriminative baseline.

Premise	Hypothesis
A smiling costumed **woman** is holding **an umbrella**	**A** happy **woman** in **a** fairy costume holds **an umbrella**
Remainder	**Bias**
A smiling costumed **woman** is holding **an umbrella** <SEP> **A** happy **woman** in **a** fairy costume holds **an umbrella**	**A** <mask> <mask> **woman** <mask> <mask> **an umbrella** <SEP> **A** <mask> **woman** <mask> **a** <mask> <mask> <mask> **an umbrella**

Table 11: Example for the data preparation for the overlap bias case.

results in good o.o.d performance and relatively small generalization gaps (row 5). This setting achieve the smallest generalization gap that still beats the discriminative baseline (row 11).

Another consideration is the additional training time incurred by two phases of training. If we skip the generative training phase and directly train with the discriminative objective, we lose a bit in terms of test performance but maintain a good o.o.d performance, resulting in a medium-size generalization gap (row 9).

The model on row 9 shows comparable performance to the one in row 5, with a slight performance drop and a larger standard derivation. In practice, that model demonstrated slight instability and performed worse on the test and hard test sets than the model on row 6, showing that the initial generative training phase may allow the model to generalize better.

A.4 Data preparation for overlap bias

Table 3 shows an example for how a P, H pair is transformed to R, B which are used as an input to the model in Section 4.3.

B Gradient Attributions Example

Table 13 gives qualitative examples for the phenomenon in Section 6.

B.1 Hyperparameters and Training Details

Table 12 shows the hyperparameters for the models used throughout the paper. We experimented with word dropout values of: $0.01, 0.1, 0.3, 0.5$, weight decay values of: $0.001, 0.01, 0.1, 1$, learning rate values in the range: $[10^{-6}, 10^{-4}]$, and maximum number of $5, 10, 20$ and 100 epochs. The values that achieved the best accuracy on the validation set appear in the table. All other hyperparameters are the default ones in Wolf et al. (2019). Where mean

Model	**Learning rate**	**No. of epochs**	**Word dropout**	**Weight decay**
Discriminative / Hypothesis-only	10^{-5}	20	–	–
Generative	10^{-5}	20	–	–
Fine-tuning	$5 \cdot 10^{-6}$	5	0.1	0.1

Table 12: Hyperparameters for models. All of the models used early stooping of 3 epochs without improvement.

Premise	**Hypothesis**	**Label**
a woman in a black shirt looking at a bicycle .	a woman dressed in black shops for a bicycle .	entailment
a black man in a white uniform makes a spectacular reverse slam dunk to the crowd ' s amazement.	the man is asian	contradiction

Table 13: Gradient attributions example. Green/red show positive/negative attributions.

and standard deviation is specified, we calculate those values over 3 runs, each with a different seed. Otherwise, those are the results of only one run.

Each experiment was performed on one or two NVIDIA RTX 2080 Ti GPUs. Training takes about 6–7 hours for discriminative models, 7–8 hours for generative models, and 15–20 hours for the discriminative fine-tuning step. Discriminative BERT/BART models have 109M/140M parameters, while generative BERT/BART models have 247M/139M parameters.

The experiment in Section 8 were preformed using NVIDIA A100s cards. The statistics for those experiments are presented in Table 14. All experiments used a batch size of 32, except for the model-tuned T5-XL and T5-XXL, which were trained with batch sizes of 16 and 8 respectively . For a fair comparison, we used T5.1.1 "LM Adapted" checkpoints,[6] which are compatible with both model-tuning and prompt-tuning. For prompt-tuning, we used 20 additional tokens, resulting in <100K trainable parameters even for the largest 11B model.

[6]https://github.com/google-research/text-to-text-transfer-transformer/blob/main/released_checkpoints.md#lm-adapted-t511lm100k

	Model	Number of Parameters (approx.)	Training Time (hours)	Number of GPUs For Training
Model-Tuning	Small	60M	2	1
	Base	220M	5	2
	Large	770M	10	2
	XL	2.8B	24	4
	XXL	11B	48	8
Prompt-Tuning	Small	60M + 10K	2	1
	Base	220M + 15K	4	1
	Large	770M + 20K	6	1
	XL	2.8B + 40K	15	2
	XXL	11B + 80K	20	4

Table 14: T5 model statistics. For the number of parameters for prompt-tuning, $X + Y$ means that the model have X fixed parameters, and additional Y learnable parameters are used for prompt-tuning.