Low-Rank Adaptation Improves Adversarial Robustness in Fine-Tuned Language Models

Litan Li
UT Austin

Will Sullivan
UT Austin

Abstract

Language models often exploit dataset artifacts which are spurious correlations that inflate in-domain accuracy while degrading robustness. Mitigating these artifacts in large models typically requires expensive retraining, motivating methods that work under strict compute limits. Using ELECTRA-small on a single GPU, we evaluate three approaches: heuristic data inoculation, ensemble-based debiasing, and Low-Rank Adaptation (LoRA). LoRA fine-tuning matches or exceeds full fine-tuning on MNLI, approaches SQuAD performance, and substantially improves robustness to universal adversarial triggers. Ensemble-based debiasing offers similar trigger-word gains with modest accuracy tradeoffs. In contrast, heuristic inoculation nearly eliminates syntactic heuristic biases (HANS) but offers limited improvements on other artifact types. Our results show that no single method uniformly mitigates all biases, but LoRA provides the strongest overall robustness-accuracy tradeoff under tight compute budgets.

1 Introduction

Large language models (LLMs) rely on modeling the probability density of data observed in the real world. Unlike physical equations which describe causal relationships, statistical models produced via maximum likelihood estimation (MLE) often exploit spurious correlations to "game" the minimization of the loss function McCoy et al. [2019]. While effective for optimizing training objectives, this behavior leads to brittleness against adversarial examples and poor generalization on held-out samples.

This generalization gap is often addressed by scaling model size or data volume, but modern LLMs have saturated web-scale data and require immense compute resources for fine-tuning. For researchers and practitioners with limited resources (e.g. a single GPU), efficiently improving robustness remains an open challenge.

In this work, we explore techniques to improve model generalization without incurring high computational costs. We focus on two distinct classes of artifacts: *structural heuristics* (e.g. lexical overlap bias) and *lexical triggers* (Universal Adversarial Triggers). Using ELECTRA-small as a testbed, we compare three mitigation strategies:

1. **Heuristic Inoculation:** Augmenting training data with counter-examples to break syntactic pattern matching.
2. **Ensemble-Based Debiasing:** Training a "debiased" model to learn the residual of a hypothesis-only "expert" model.
3. **Low-Rank Adaptation (LoRA):** A parameter-efficient fine-tuning method we hypothesize may act as a regularizer against overfitting to artifacts.

We find that all three strategies mitigate dataset artifacts, though they differ significantly in their impact on in-domain performance. We observed a clear robustness-accuracy trade-off with ensemble-based debiasing, where increasing the bias weight (λ) to improve trigger robustness directly degraded MNLI

validation accuracy. In contrast, Heuristic Inoculation and LoRA emerged as Pareto improvements: Inoculation neutralized syntactic heuristics without compromising in-domain accuracy, while LoRA fine-tuning simultaneously improved resistance to adversarial triggers and exceeded the baseline's validation performance. This suggests that while explicit debiasing penalties can harm general learning, methods that enrich the training data or regularize optimization (like LoRA) can achieve robustness as a co-benefit rather than a trade-off.

We begin by establishing full fine-tuning baselines for MNLI and SQuAD (§2–3). We then analyze the dataset artifacts present in MNLI, focusing on syntactic heuristics and universal adversarial triggers (§4). Next, we evaluate three families of techniques to mitigate these artifacts under a single-GPU compute budget: (1) heuristic inoculation via HANS augmentation (§5.1), (2) ensemble-based debiasing using a hypothesis-only expert model (§5.2), and (3) parameter-efficient fine-tuning through LoRA adapters (§5.3). We compare these approaches with respect to clean accuracy, HANS robustness, and trigger-word robustness before concluding with recommendations for practical, resource-constrained fine-tuning (§6).

2 Model

2.1 Heads

We used the ELECTRA small model, which includes a 14 million parameter pretrained discriminator plus a randomly initialized head for post-training tasks. The head is a binary or multi-class classifier for sequence classification tasks such as natural language inference (NLI). For MNLI (Wang et al. [2019]) the head is a 3-way classifier that returns logits for the three label classes: entailment, neutral, and contradiction. For question answering tasks such as SQuAD (Rajpurkar et al. [2016]), the head is simply a linear layer that outputs the start and end logits corresponding to ground truth indices of the answer spans.

2.2 Loss

For all experiments, we optimize the standard cross-entropy loss over raw logits. Notably, DRiFt-style debiasing modifies only the logit composition before the loss is computed, while LoRA modifies the parameterization of the logits. Thus, all methods remain comparable under a unified loss function.

3 Baseline Models

3.1 With recommended hyperparameters

Using the recommended fine-tuning recipe from Clark et al. [2020], we performed supervised fine-tuning (SFT) on ELECTRA-small with MNLI for 5 epochs and saved the checkpoint with the best validation accuracy. We repeat the above to create the SQuAD Baseline checkpoint with the best validation exact match.

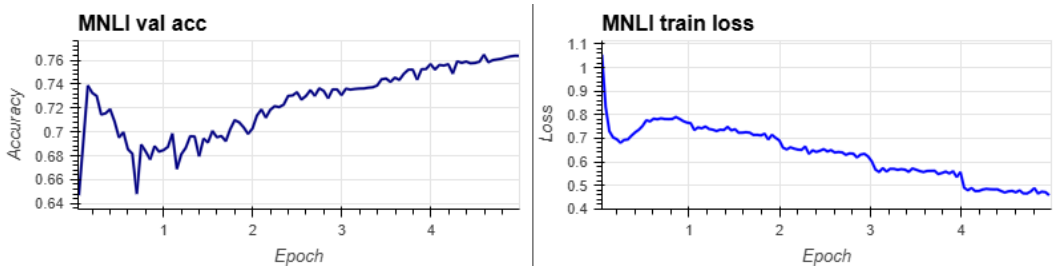

Figure 1: MNLI validation accuracy (left) and training loss (right) vs. number of epochs on the MNLI dataset.

The early phase of MNLI fine-tuning showed a sharp change in accuracy and training loss, followed by stable improvement. The sharp decrease can be attributed to rapid overfitting of the randomly initialized head to the first few batches. This does not necessarily lead to improvements in validation

metrics since it is overfitting to training examples, which can be shown in the corresponding decrease in validation accuracy. However, as the weight values of the classification head stabilized, training proceeded smoothly with gradually improving generalization performance (increasing validation accuracy) and learning (decreasing training loss) on the MNLI dataset. We achieve 76% validation accuracy after 5 epochs. Although extrapolation of the validation accuracy suggests further improvement beyond 5 epochs, reaching asymptotic limits on a singular dataset is not necessary for this study.

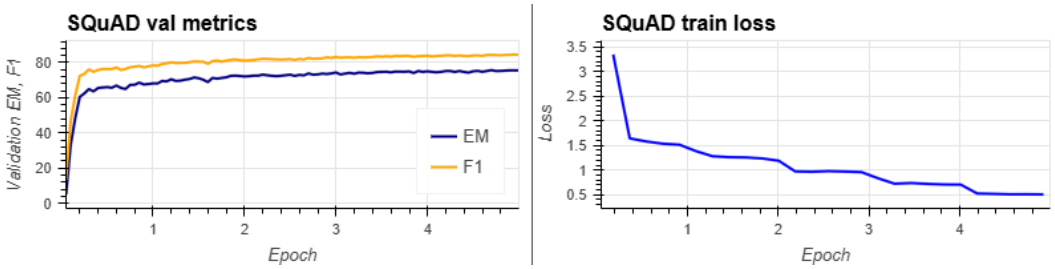

Figure 2: SQuAD validation exact match and F1 (left), training loss (right) vs. number of epochs on the SQuAD v1 dataset.

Figure 2 shows no instability and gradual learning over time. The sharp rise in validation metrics during the early iterations suggest rapid learning of the head weights, followed by steady generalization improvement. We achieve 84% F1 and 75% exact match after 5 epochs.

3.2 Tweaked hyperparameters

Table 8 from the ELECTRA paper Clark et al. [2020] reports that ELECTRA-small achieved 79.7% on MNLI after 3 epochs of training, which roughly represents a 5% gap from our attempt at reproducing their results in the previous section. We doubled the batch size from their recommended value of 32 to 64, taking half as many gradient steps per epoch but with better estimated gradients at each step. Figure 3 shows that after only 1.5 epochs of training we recovered a validation accuracy of 78.5%, commensurate with Clark et al.'s results. We refer to this checkpoint as the **MNLI Baseline**, which represents ELECTRA-small's baseline SFT performance on MNLI. Our only explanation for this discrepancy, short of code bugs, is that taking gradient steps with more accurate gradient estimates perhaps helped the randomly initialized head converge much faster, since we no longer see the initial dip.

Similarly we doubled the batch size for SQuAD. To our knowledge Clark et al. [2020] only reported SQuAD validation accuracy on ELECTRA-large. Nonetheless we found that doubling the batch size obtained better validation exact match (76.5%) and F1 scores (84.7%) at 1.5 epochs. We refer to this checkpoint as the **SQuAD Baseline**.

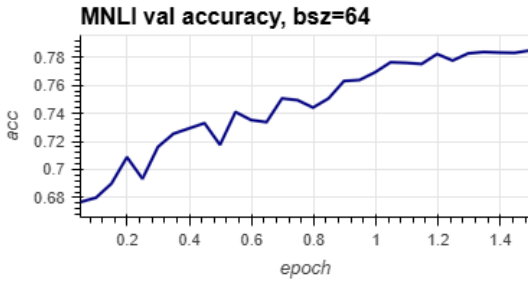

Figure 3: MNLI Baseline with batch size 64 reaches 78.5% validation accuracy after 1.5 epochs, commensurate with Clark et al. [2020].

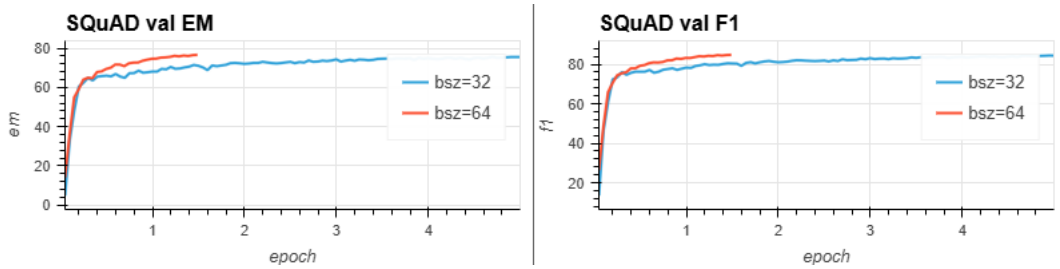

Figure 4: SQuAD Baseline with batch size 64 reaches 76.5% exact match and 84.7% F1 at 1.5 epochs. Both are higher than at 5 epochs with batch size 32 (75%, 84%).

Heuristic	E Acc.	¬E Acc.
Lexical Overlap	98.7	1.6
Subsequence	100	0.7
Constituent	100	0.9

Table 1: MNLI Baseline model accuracy on the HANS evaluation set, broken down by heuristic type.

4 Dataset artifacts

4.1 Heuristics

To analyze the MNLI Baseline's shortcomings, we evaluated the model against a popular adversarial challenge set, the Heuristic Analysis for NLI Systems (HANS) from McCoy et al. [2019]. The HANS dataset is designed to test whether a model relies on brittle heuristics such as lexical overlap, subsequence, and constituent patterns. Table 1 shows heuristic example accuracies from running the HANS evaluation dataset through the MNLI Baseline model. Note that the HANS dataset uses two classes (entailment and non-entailment) vs three classes (entailment, contradiction, and neutral).

Heuristic entailed examples use heuristics to lead the model to the correct answer, while heuristic non-entailed examples lead the model to the incorrect answer. Table 2 lists a heuristic example from each category.

HANS reveals that ELECTRA-small relies heavily on lexical overlap, subsequence, and constituent heuristics: it excels on heuristic-entailed examples and fails almost completely on counter-heuristic cases. This confirms the presence of strong dataset artifacts in MNLI.

4.2 Universal adversarial triggers

Wallace et al. [2021] defines a *universal adversarial trigger* (UAT) as an "input-agnostic sequence of tokens that trigger a model to produce a specific prediction when concatenated to any input from a

Heuristic	Premise	Hypothesis
Lexical Overlap	The actors avoided the bankers.	The bankers avoided the actors.
Subsequence	The lawyers knew the professors mentioned the presidents.	The lawyers knew the professors.
Constituent	Whether or not the professor danced, the student waited.	The professor danced.

Table 2: Heuristic examples by category.

dataset". This can result in wildly low accuracy, such as the SNLI entailment accuracy dropping from 89.94% to 0.55%, as noted in Wallace et al. [2021].

Wallace et al. [2021] efficiently finds the most potent UATs through a white box attack that involves iteratively replacing tokens in the trigger to minimize loss. This white box method requires full access to the model's internals which is not always available. We chose to analyze UATs via black box attack to generalize our findings for closed-source models.

To avoid the computational expense of a full vocabulary black box attack, we first calculated word-label correlations for each word in the MNLI dataset's vocabulary as in Poliak et al. [2018]. That is, we calculated the conditional probability of a specific label given a word.

$$P(label \mid word) = \frac{N(word, label)}{N(word)} \tag{1}$$

This filtering approach is similar to that from Gururangan et al. [2018] where they computed point-wise mutual information (PMI) between each word and class.

$$PMI(label, word) = \log \frac{P(label \mid word)}{P(label)} \tag{2}$$

We focused purely on posterior probability since MNLI is a balanced training set and because models trained with cross-entropy loss optimize posterior probability which should make raw conditional probability a better proxy for learned artifacts. To remove low frequency noise from the distribution we filtered with a minimum word count per label of 20 and minimum probability threshold of 0.7. This allowed us to filter the entire dataset vocabulary into a short list of words that have suspicious biases towards a certain label. We found 42 contradiction, 2 entailment, and 14 neutral examples that met this criteria. Table 3 list the top five potential UATs per label.

Label	**Word**	**Prob**	**Count**
Contradiction	nah	0.920	25
	indefinitely	0.912	34
	whatsoever	0.892	111
	devoid	0.871	124
	unremarkable	0.867	33
Entailment	supposedly	0.735	34
	ranging	0.714	21
	...	...	...
	...	...	...
	...	...	...
Neutral	vampire	0.848	46
	vampires	0.826	69
	demons	0.798	94
	purposely	0.773	22
	regretted	0.773	22

Table 3: Top Conditional Probabilities for Each NLI Label

The entailment label had surprisingly few potential UATs that met the frequency criteria. This might indicate that entailment is often achieved through generic words and approximations (e.g. premise: "They drove a Ford" ; hypothesis: "They drove a car") as described in Gururangan et al. [2018]. Additionally, the primary artifacts for the entailment label tend to be heuristic-based such as high word overlap and short hypotheses as noted in McCoy et al. [2019]. Therefore, the entailment label lacks specific UATs such as the characteristic negative UATs for the contradiction label (e.g. nah, unremarkable). The white box method used in Wallace et al. [2021] is superior in this regard since it uses gradients to find the sequence that will maximize the model's output confidence for a given label and does not rely on statistical frequency.

To supplement our list of 58 potential UATs, we included commonly cited UATs for NLI from previous research including Gururangan et al. [2018], Wallace et al. [2021], and Poliak et al. [2018].

We executed the adversarial attack with the list of potential UATs in a similar manner to Wallace et al. [2021] by prepending a single UAT to each hypothesis in the MNLI validation dataset in the form *{word} and {hypothesis}*. The MNLI Baseline model's performance on these modified datasets is compared to the MNLI Baseline's performance on the MNLI validation dataset in Figure 5 (displaying the three lowest accuracies for each label).

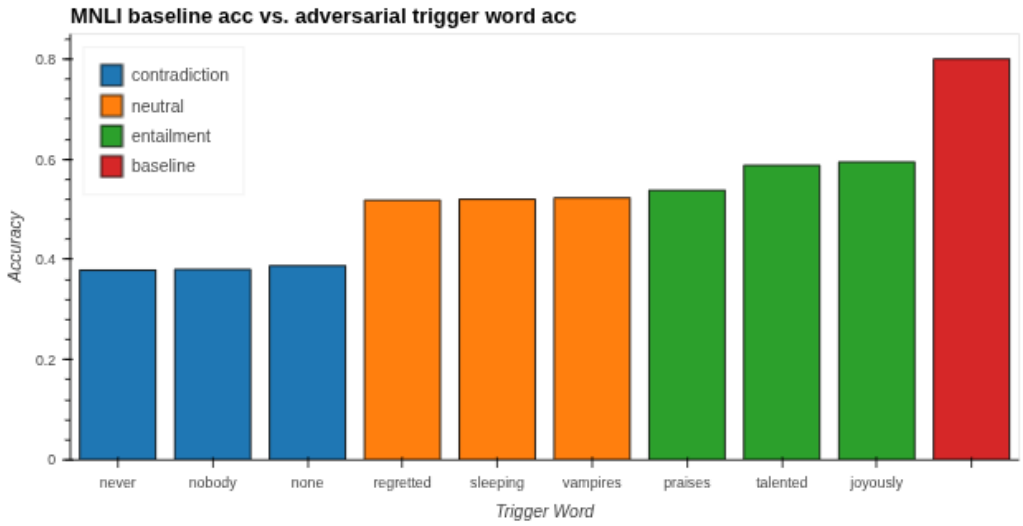

Figure 5: Contradiction UAT inoculation results in an accuracy just above random chance.

While the reduction in model accuracy is not nearly as pronounced as the UAT adversarial attacks performed in Wallace et al. [2021], the model achieves just above random chance accuracy (33.3%) when evaluating a dataset inoculated with contradiction UATs.

To capture only the most severe UATs, we filtered our UAT list to include only words that induced lower than 60% accuracy. The final list is summarized in Table 4.

Label	**UATs (Acc. $\leq$ 60%)**
Contradiction	*nobody, no, not, never, nothing, none, neither, nor, impossible, unlikely, false, nope, refused, refuses, without, lack, sad, scared, hate, ugly, dreadful, pathetic, ridiculous, meaningless, terrible, awful, boring, stupid, horrible, disgusting, sucks, worthless, useless, devoid, barren, uninhabited, deserted, empty, silent, motionless, uninterested, unimportant, irrelevant, unrelated, uninteresting, unremarkable, drab, dull, indefinitely, plumber, disbanded, universally, frowned, perfectly, utterly, untouched, skip, completely, outlawed*
Neutral	*vampires, vampire, demons, businessman, google, wealthiest, humour, comedy, sleeps, sleeping, regretted, secretly, reluctant, outraged*
Entailment	*joyously, praises, talented*

Table 4: The contradiction class is the dominant UAT label.

The domination of contradiction UATs in Table 4 is consistent with Gururangan et al. [2018] and Poliak et al. [2018], which establish that negation (e.g. not, neither, nor) and positive word antonyms (e.g. horrible, disgusting, meaningless) are heavily exploitable, while entailment/neutral labels typically require synonyms or paraphrasing.

5 Approaches

5.1 Heuristic inoculation

5.1.1 Method

To correct the MNLI Baseline model's spurious heuristic correlations that the HANS dataset brought to light, we used the heuristics train dataset, developed by McCoy et al. [2019], which contains over 30,000 heuristic examples spread between lexical overlap, subsequence, and constituent patterns. The train dataset does not contain examples used in the HANS dataset to avoid invalidating the evaluation dataset. We shuffled the heuristics train dataset into the MNLI train dataset and mapped all "non-entailment" labels in the heuristics dataset to "neutral" since the heuristics dataset only contains "entailment" and "non-entailment" labels. We then trained an ELECTRA-small model on this new inoculated dataset.

5.1.2 Results

Table 5 shows heuristic example accuracies from running the HANS evaluation dataset through the ELECTRA-small model trained on the inoculated dataset. The jump in heuristics non-entailed accuracies (examples where the heuristic incorrectly points to entailment) proves that the model no longer relies on brittle heuristic correlations. This effectively reproduced the results in McCoy et al. [2019] for a BERT model trained on a MNLI dataset augmented with HANS examples.

Heuristic	E Acc.	¬E Acc.
Lexical Overlap	100	99.8
Subsequence	100	100
Constituent	100	100

Table 5: MNLI Baseline model accuracy on the HANS evaluation set, broken down by heuristic type.

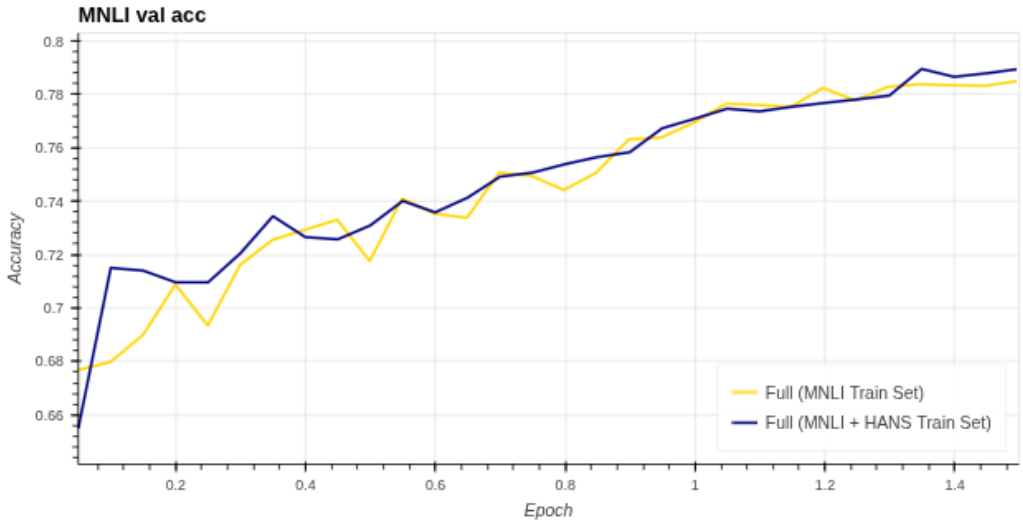

Figure 6: ELECTRA-small model trained on MNLI train dataset with HANS inoculation achieves slightly higher accuracy than the base MNLI dataset.

Figure 6 shows that the ELECTRA-small model trained for 1.5 epochs on the MNLI dataset with HANS inoculation performs slightly better than the MNLI Baseline model. The boost from 78.5% to 78.9% reflects the similar boost observed to BERT from 84.1% to 84.4% in McCoy et al. [2019]. This slight increase in accuracy could be attributed to the additional training volume provided by the 30,000 heuristic examples. The improved heuristic accuracy comes at no cost to the overall MNLI validation accuracy.

5.2 Ensemble-based debiasing

5.2.1 Method

A common method to improve a model's performance in the face of potential adversarial attacks is to train the model on adversarial datasets as in Zhou and Bansal [2020]. We chose to correct the UAT bias via a less direct method: ensemble-based debiasing. Building upon He et al. [2019], we trained a known biased model by removing the premises from the MNLI train dataset. This hypothesis-only model learns to rely on hidden dataset biases to arrive to the correct answer. The biased model used in ensemble-based debiasing is often called the artifact expert because it is designed to be an "expert" at exploiting shortcuts and artifacts in the training data. We will refer to the hypothesis-only model as the expert model.

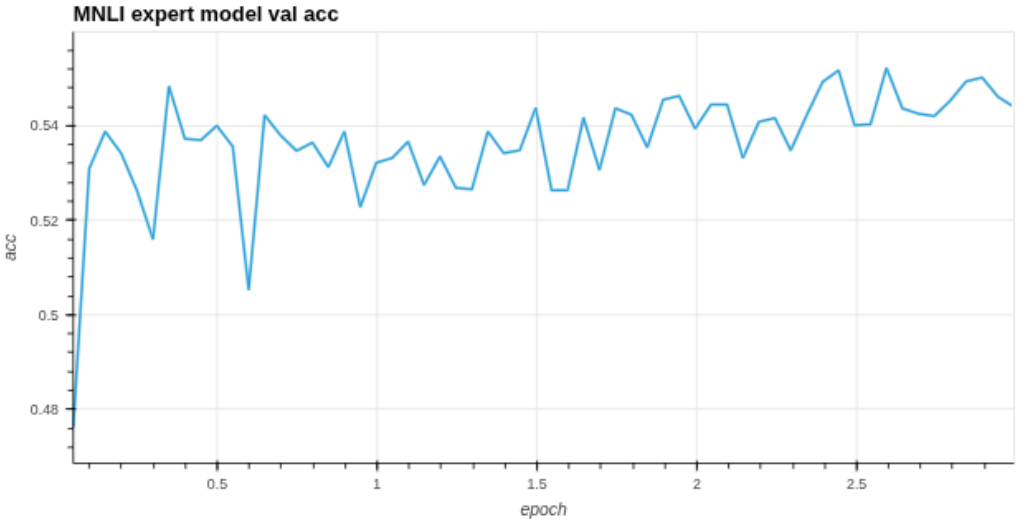

Figure 7: Expert model performance far exceeds random chance.

As shown by Figure 7, the expert model performs well-above random chance accuracy of 33.3%. We achieved a peak validation accuracy of roughly 55.5% after 5 epochs of training which reflects the 53.9% accuracy achieved by Gururangan et al. [2018] using a simple text-classifier.

Following He et al. [2019], we adopt a residual debiasing framework in which a biased expert model captures dataset artifacts and second model learns only the remaining signal. Let f_s denote the biased hypothesis-only model trained on hypothesis-only inputs $I(x)$, and let f_d denote the debiased model trained on hypothesis/premise inputs x. Since f_s has access only to the hypothesis, it learns correlations associated with annotation artifacts, lexical heuristics, and other spurious signals. We first train f_s by minimizing the loss $L(f_s(I(x)), y)$ and freeze its parameters θ^*. We add an additional hyperparameter, λ, to scale the biased model's logits prior to addition. The final predictor is the additive model

$$f^*(x) = \lambda f_s(I(x); \theta^*) + f_d(x; \phi).$$

We then train f_d by minimizing

$$\min_{\phi} \; L(\lambda f_s(I(x); \theta^*) + f_d(x; \phi), y),$$

so that f_d learns only the residual not already captured by the biased model. At test time, following He et al. [2019], we discard the expert model and evaluate using f_d alone. This forces the model to avoid relying on shortcuts learned by the hypothesis-only expert during training.

He et al. [2019] calls their hypothesis-only expert debiased model DRiFt-HYPO. To differentiate models, we will call our debiased models DRiFt-λ where λ is the value of bias weight λ.

To select a hypothesis-only expert model checkpoint to use as our biased model, we evaluated MNLI validation accuracy and HANS accuracy for a few epochs. As expected, the HANS dataset has minimal effect on the expert models and classification is essentially random over the "entailment" and

"non-entailment" labels. These results reflect those obtained in He et al. [2019] (labeled "HYPO"). He et al. [2019] trained their BERT HYPO model for 4 epochs. We chose epoch 3 for our ELECTRA-small expert model, as recommended by Clark et al. [2020].

Epochs	MNLI Acc.	Lexical Overlap		Subsequence		Constituent	
		E	¬E	E	¬E	E	¬E
0.5	53.7	55.5	43.7	59.4	40.3	60.7	33.6
1.0	52.3	63.0	23.7	62.0	20.0	64.6	18.2
1.5	53.5	57.0	40.6	58.1	28.3	61.1	32.0
3.0	55.0	50.8	50.3	55.7	54.8	51.4	49.5
5.0	55.5	56.0	42.4	58.1	45.8	56.2	42.5
HYPO	52.5	52.6	44.4	54.5	44.3	45.6	16.7

Table 6: Comparison of HANS Heuristic F1: Expert model epochs

5.2.2 Results

We trained ELECTRA-small models with four different bias weights. Figure 8 shows that validation accuracy decreases slightly, proportional to the magnitude of the bias weight.

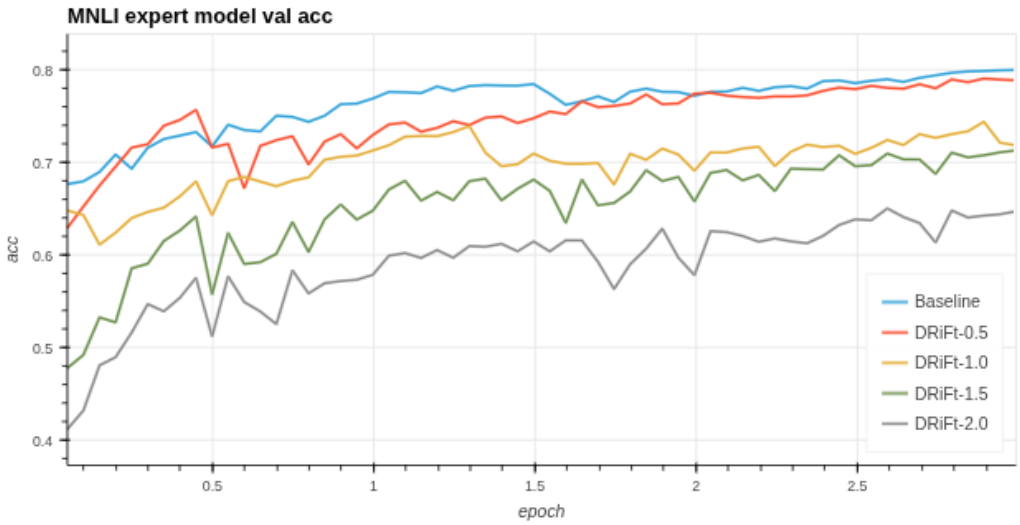

Figure 8: Evaluation accuracy reduces proportional to bias weight for ensemble-debiased models.

To analyze targeted performance gains, we first conducted the same UAT adversarial attack that we performed on the MNLI Baseline model. We calculated relative gain by taking into account the lower ensemble model validation accuracy. To quantify how a model relies on spurious cues, we define a *retention* metric. Let $\mathcal{D}_{\text{trigger}}$ denote the subset of the evaluation set containing the targeted UATs, and let $\mathcal{D}_{\text{eval}}$ denote the full validation set. Then the retention R is defined as the ratio of accuracies:

$$R = \frac{\text{Acc}(\mathcal{D}_{\text{trigger}})}{\text{Acc}(\mathcal{D}_{\text{eval}})} \tag{3}$$

where $\text{Acc}(\cdot)$ returns the accuracy of the model on a given dataset. Intuitively, R measures the fraction of performance retained on the trigger subset relative to the overall model performance.

The top five improved UAT accuracies for DRiFt-0.5 are listed in Table 7. The ensemble model clearly improves UAT robustness compared to the MNLI Baseline model. Table 8 summarizes relative difference and retention rate for each ensemble model.

UAT	**Rel.** Δ	**Model Acc.** (Ret.)	**Base Acc.** (Ret.)
unremarkable	+17.39	66.36 (84.1)	53.37 (66.7)
unimportant	+13.88	62.73 (79.5)	52.50 (65.6)
refused	+12.48	48.78 (61.8)	39.48 (49.4)
indefinitely	+11.38	61.33 (77.7)	53.08 (66.4)
motionless	+11.04	55.64 (70.5)	47.58 (59.5)

Table 7: Top UAT improvements, sorted by relative robustness.

Model	**MNLI Acc.**	**Rel. Trigger** $\overline{\Delta}$	$\overline{\textbf{Retention}}$
Baseline	80.0	—	64.3
DRiFt-0.5	78.9	+4.3	68.6
DRiFt-1.0	74.4	+5.8	70.1
DRiFt-1.5	71.3	+7.9	72.2
DRiFt-2.0	64.7	+7.2	71.5

Table 8: Ablation Study of Ensemble Debiasing (λ Weight vs. Robustness)

We next evaluated the ensemble models on the HANS dataset. The ensemble models show far lower improvement over MNLI Baseline as compared to DRiFt-HYPO (He et al. [2019]). We attribute this disparity to either ELECTRA-small's lower capacity compared to BERT-base (14M vs. 110M parameters) or the difference between pre-training objectives (Clark et al. [2020]). Instead of input-masking, ELECTRA-small pre-training involves substitution with plausible alternatives via a generator and predicting whether each token was placed by the generator. Since ELECTRA-small trains on 100% of its inputs tokens, it is highly efficient compared to BERT. The combination of a lower capacity and altered pre-training may reduce the efficacy of a hypothesis-only ensemble-debiased model on the HANS dataset. Table 9 compares F1 scores (averaged over the three heuristic categories) by model.

(a) Entailment (E)

Model	Lexical Overlap	Subsequence	Constituent
Baseline	66.5	66.8	66.8
DRiFT-0.5	66.0	67.2	66.5
DRiFT-1.0	65.9	67.1	65.7
DRiFT-1.5	65.6	67.2	66.8
DRiFT-2.0	64.6	68.3	66.3
DRiFt-HYPO	84.7	69.0	72.7

(b) Non-Entailment (¬E)

Model	Lexical Overlap	Subsequence	Constituent
Baseline	3.2	1.5	1.7
DRiFT-0.5	4.4	5.3	9.3
DRiFT-1.0	2.2	4.3	8.6
DRiFT-1.5	7.0	12.6	9.1
DRiFT-2.0	8.3	20.2	14.4
DRiFt-HYPO	79.8	23.7	40.8

Table 9: Per-class F1 scores for HANS categories.

Improving model robustness must be weighed against validation accuracy reduction. Taking this into account, a bias weight of 0.5 seems to have the best robustness vs. accuracy tradeoff. It may be beneficial to test smaller bias weights to determine the optimal ratio.

5.3 Low-rank adaptation

Low-rank adaptation (LoRA) Hu et al. [2021] is a very popular technique for parameter-efficient fine-tuning (PEFT). The training procedure for our Baseline models trains all the model weights, including the discriminator and the head. We refer to this as full fine-tuning in this section. In contrast, LoRA only trains adapter layers represented as deltas on top of the original weights, where the deltas are outer products of two low rank matrices. During LoRA fine-tuning all the original model weights are frozen while only the adapter weights are trained. This increases the FLOPS and memory requirement on the forward pass, but drastically reduces the FLOPs and memory required by the optimizer states and the backward pass.

5.3.1 Method

Following Schulman and Lab [2025] we fine-tuned the pretrained ELECTRA-small model with LoRA. We used a learning rate (3e-3) that is 10 times the recommended learning rate for ELECTRA-small fine-tuning from Clark et al. [2020]. Sweeping the LoRA rank from 1 to 256, we used a value of 32 for alpha and a batch size of 64 for all runs. Following LoRA best practices we excluded the classifier head and QA outputs head, embedding layers, and layer norms, whilst applying the adapter to the self attention (query, key, value projections) and dense linear layers in the pre-trained discriminator.

5.3.2 Results

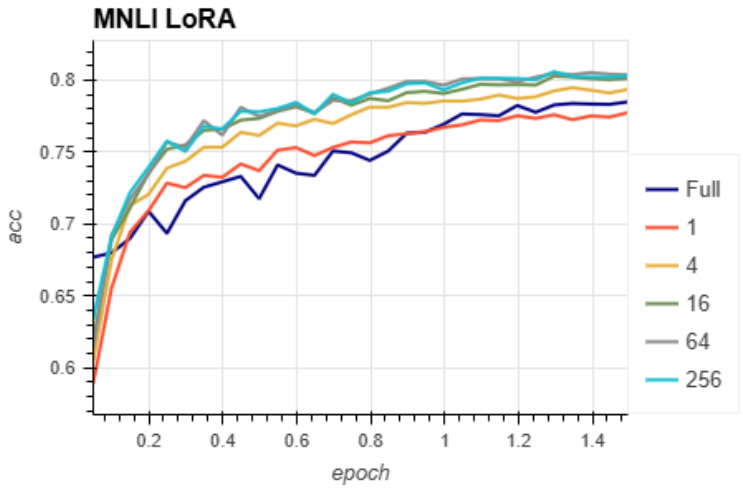

Figure 9: MNLI validation accuracy for LoRA rank sweep across 1 to 256, compared to full fine-tuning. We reach 80% accuracy after 1.5 epochs with LoRA rank greater or equal to 16, 1.5% higher than the MNLI Baseline (78.5%).

Figure 9 shows that with enough adapter capacity to learn the MNLI training data (LoRA rank 4 or greater), LoRA fine-tuning outperforms full fine-tuning at 1.5 epochs. In contrast, when each adapter matrix was rank 1, the adapters did not have enough parameters for effective learning. We also note that LoRA fine-tuning begins at a lower accuracy than full fine-tuning, possibly because the adapters are initialized with random weights, which means that the forward pass during the first evaluation step contained not only a lightly-trained head, but also lightly-trained adapter layers. Extra lightly-trained parameters means the network needed more of a warm up before significant learning occurred.

Figures 10 and 11 show SQuAD validation exact match and F1 reaching 73.6% and 82.7% with rank-64 adapters, within 3% exact match and 2% F1 of SQuAD full fine-tuning. This is typical since

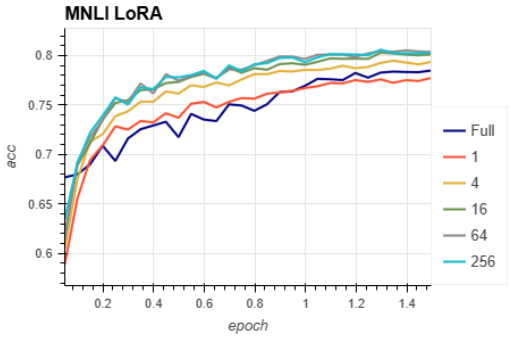

Figure 10: MNLI validation accuracy. We reach 80% accuracy after 1.5 epochs with LoRA rank ≥ 16.

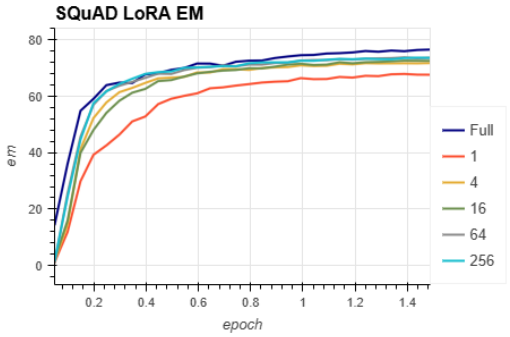

Figure 11: SQuAD exact match. We reach 71.7% after 1.5 epochs with LoRA rank > 1.

full fine-tuning unfreezes the entire model during post-training, which has many more parameters. A noticeable improvement between LoRA rank 1 to 16 suggests that a minimum capacity must be reached in order for learning to be effective, which was also shown in Schulman's work. After LoRA rank 16, negligible improvement was shown by increasing the size of the LoRA adapters via higher rank for both datasets.

To gauge the robustness of the LoRA models, we conducted the same HANS and UAT black box attack that we performed on the ensemble models. Table 10 and Table 11 summarize the results.

UAT	**Rel.** Δ	**Model Acc.** (Ret.)	**Base Acc.** (Ret.)
unrelated	+32.7	74.8 (93.0)	47.4 (60.4)
disbanded	+31.6	67.9 (84.5)	41.5 (52.9)
frowned	+26.3	69.9 (87.0)	47.6 (60.7)
outlawed	+17.4	60.0 (74.6)	44.9 (57.2)
refused	+17.2	54.3 (67.5)	39.5 (50.3)

Table 10: Top UAT improvements, sorted by relative robustness for LoRA-64

Model	**MNLI Acc.**	**Rel. Trigger** $\overline{\Delta}$	$\overline{\textbf{Retention}}$
Baseline	80.0	—	64.3
LoRA-4	79.4	+7.1	72.6
LoRA-16	80.1	+10.3	75.9
LoRA-64	80.4	+9.6	75.1
LoRA-256	80.3	+11.3	76.9

Table 11: Ablation Study of Ensemble Debiasing (λ Weight vs. Robustness)

The LoRA models significantly improve UAT robustness with over a 30% increase in relative accuracy for some UATs and an average increase around 10%. Freezing the pre-trained ELECTRA-small weights and only updating a low-rank adaptor may make it mathematically difficult for the model to learn sharp spurious weight updates. The model is forced to rely on its more robust pre-trained understanding of words like "unrelated" rather than simple lexical artifacts in the MNLI training dataset.

(a) Entailment (E)

Model	Lexical Overlap	Subsequence	Constituent
Baseline	66.5	66.8	66.8
LoRA-4	66.2	67.1	65.9
LoRA-16	66.3	66.9	65.9
LoRA-64	66.1	67.2	67.4
LoRA-256	66.4	66.9	67.1
DRiFt-HYPO	84.7	69.0	72.7

(b) Non-Entailment (¬E)

Model	Lexical Overlap	Subsequence	Constituent
Baseline	3.2	1.5	1.7
LoRA-4	1.5	7.6	6.9
LoRA-16	1.2	4.8	16.1
LoRA-64	1.6	4.8	17.7
LoRA-256	1.0	6.0	9.5
DRiFt-HYPO	79.8	23.7	40.8

Table 12: Per-class F1 scores for HANS categories.

The LoRA models improvement over MNLI Baseline on the HANS dataset subsequence and constituent patterns, but slightly worse performance on lexical overlap. Results are comparable to those from the DRiFT-λ models. We attribute this slight improvement in two of the patterns to LoRA's role as a regularizer that effectively preserves the pre-trained model's syntatic competence, preventing the model from discarding robust linguistic features for spurious artifacts during fine-tuning. In contrast, the DRiFT-λ models seek to actively unlearn artifacts vice maintaining pre-trained knowledge.

6 Related Work

LLM Fine-Tuning and Parameter Efficiency Large language model fine-tuning traditionally updates all model parameters, which is expensive in both memory and compute. Parameter-efficient fine-tuning (PEFT) techniques such as adapters, prefix tuning, and low-rank adaptation (LoRA) reduce training cost by introducing small trainable modules while freezing the base model Hu et al. [2021]. Recent work by Schulman and Lab [2025] demonstrates that LoRA adapters can match or exceed full fine-tuning performance when scaled appropriately. Our work extends these findings to a smaller model (ELECTRA-small) under strict single-GPU constraints.

Dataset Artifacts and NLI Biases The MNLI and SNLI datasets are known to contain strong lexical and syntactic artifacts that models learn as shortcuts rather than true reasoning patterns. Gururangan et al. [2018], Poliak et al. [2018] show that simple word-label correlations let a hypothesis-only classifier achieve well-above-chance accuracy. HANS McCoy et al. [2019] exposes these spurious heuristics explicitly through challenge sets designed around lexical overlap, subsequence, and constituent structure. Our artifact analysis builds directly on this line of work.

Adversarial Examples and Universal Triggers Wallace et al. [2021] show that universal adversarial triggers (token sequences prepended to any input) can collapse accuracy to nearly zero. Their white-box method uses gradients to optimize trigger tokens; we instead develop a black-box, dataset-statistics-driven trigger search suitable for closed-source or constrained-access models.

Ensemble-Based Debiasing Several works attempt to remove dataset artifacts by combining a biased model with a debiased residual model. He et al. [2019] introduced the DRiFt framework, where a hypothesis-only model captures dataset artifacts and a second model is trained to explain the residual. Our study evaluates DRiFt-style debiasing with added residual scaling in the constrained ELECTRA-small regime and compares its efficacy to LoRA-based PEFT.

7 Conclusion

We explored parameter-efficient fine-tuning via LoRA, heuristic inoculation, and ensemble-based debiasing to improve the robustness of a pre-trained language model on limited compute. Our experiments on ELECTRA-small reveal that robustness vs. accuracy is not a zero-sum game. We show that LoRA fine-tuning is highly effective at neutralizing UATs, outperforming full fine-tuning by preventing the memorization of sparse lexical artifacts. However, LoRA fails to address structural heuristics, where it performs no better than the baseline in one of the HANS heuristic categories.

In contrast, heuristic inoculation proves to be the most effective method for combating structural biases like lexical overlap, though it requires specific knowledge of the artifact to generate counter-examples. Ensemble-based debiasing offered moderate improvements across the board but introduced a trade-off with in-domain accuracy. Ultimately, we conclude that parameter-efficient methods like LoRA act as excellent regularizers for lexical robustness, but deep structural artifacts require targeted data interventions. Future work should explore combining LoRA with inoculation to achieve both lexical robustness and heuristic competence on a single-GPU budget.

References

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*, 2020.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *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, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL https://aclanthology.org/N18-2017/.

He He, Sheng Zha, and Haohan Wang. Unlearn dataset bias in natural language inference by fitting the residual, 2019. URL https://arxiv.org/abs/1908.10763.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://aclanthology.org/P19-1334/.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference, 2018. URL https://arxiv.org/abs/1805.01042.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*, 2016.

John Schulman and Thinking Machines Lab. Lora without regret. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20250929. https://thinkingmachines.ai/blog/lora/.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. Universal adversarial triggers for attacking and analyzing nlp, 2021. URL https://arxiv.org/abs/1908.07125.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*, 2019.

Xiang Zhou and Mohit Bansal. Towards robustifying NLI models against lexical dataset biases. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8759–8771, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.773. URL https://aclanthology.org/2020.acl-main.773/.