Adversarially Constructed Evaluation Sets Are More Challenging, but May Not Be Fair

Anonymous ACL submission

Abstract

Large language models increasingly saturate existing task benchmarks, in some cases outperforming humans, leaving little headroom with which to measure further progress. Adversarial dataset creation, which builds datasets using examples that a target system outputs incorrect predictions for, has been proposed as a strategy to construct more challenging datasets, avoiding the more serious challenge of building more precise benchmarks by conventional means. In this work, we study the impact of applying three common approaches for adversarial dataset creation: (1) filtering out easy examples (AFLite), (2) perturbing examples (TextFooler), and (3) model-in-the-loop data collection (ANLI and AdversarialQA), across 18 different adversary models. We find that all three methods can produce more challenging datasets, with stronger adversary models lowering the performance of evaluated models more. However, the resulting ranking of the evaluated models can also be unstable and highly sensitive to the choice of adversary model. Moreover, we find that AFLite oversamples examples with low annotator agreement, meaning that model comparisons hinge on the examples that are most contentious for humans. We recommend that researchers tread carefully when using adversarial methods for building evaluation datasets.

1 Introduction

Large-scale language models have attained strong performance across a variety of language understanding tasks, including question-answering, natural language inference (NLI), coreference resolution and paraphrase identification. Standard benchmarking tasks such as SQuAD (Rajpurkar et al., 2016; Lee et al., 2020) and multi-task benchmarks such as GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) have seen models attain scores higher than humans. This has left little headroom with which to measure further improvements in models and progress in NLP.

Prior work such as Le Bras et al. (2020) and Nie et al. (2020a) have proposed to construct more challenging datasets adversarially, either by only selecting examples that a given model predicts incorrectly, or by constructing new examples to deliberately stump a model. Both of approaches aim to raise the difficulty of task datasets by leveraging highly capable models (known as the *adversary model*) to assist with example selection or creation. However, one potential issue is that an adversarially constructed dataset that targets a specific model may bias the resulting data, creating datasets that are unduly challenging for one class of models but not others (Bowman and Dahl, 2021).

In contrast to other work focused on adversarial dataset creation for training (Wallace et al., 2021) or training and evaluation data (Le Bras et al., 2020; Nie et al., 2020b), we focus solely on evaluation data, and whether the choice of adversary model can introduce unwanted biases into an evaluation dataset. Ideally, an adversarially created dataset should be more difficult for all models, regardless of the choice of the adversary. In this work, we investigate three different approaches to create more challenging task evaluation datasets using adversary models: (1) *adversarial filtering*, which filters out examples from a static dataset that are identified to be easy for a given adversary model, (2) *adversarial perturbation*, wherein examples are modified to reduce performance of an adversary model, and (3) model-in-the-loop *adversarial data collection*, where human annotators interactively create examples that stump an adversary model.

For adversarial *filtering*, we study AFLite (Sakaguchi et al., 2020; Le Bras et al., 2020), an algorithm that identifies challenging subsets of an existing dataset. For adversarial *perturbation*, we evaluate TextFooler (Jin et al., 2019), a popular method for adversarially perturbing examples to

lower performance on a target model via word substitution. We apply both AFLite and TextFooler in experiments across four English-language NLP datasets and 18 different models to study the interaction between the choice of adversary model and the evaluation and ranking of systems on the resulting dataset. For adversarial data collection, we evaluate a range of models against two adversarially collected datasets: ANLI (Nie et al., 2020a) and AdversarialQA (Bartolo et al., 2020).

We find that all three classes of methods do result in more challenging evaluation datasets, but with some notable drawbacks. For both adversarial filtering and adversarial perturbation, the general outcome is to lower performance across the board, with stronger adversary models leading to more challenging subsets of examples. However, with both methods, the relative order of model performance is not preserved, with large random variation in model ranks as stronger adversaries are used. Performance on the resulting datasets is also much worse if the evaluated and adversary models are derived from same pretrained model, which can lead to the difficulty of the adversarially constructed dataset being overstated. Furthermore, adversarial filtering tends to oversamples examples with low annotator agreement, which means that the selected examples are often contentious even for human annotators. On the other hand, TextFooler perturbations can introduce errors and lead to label flips, and the distortion in resulting model rankings is generally larger for TextFooler than for AFLite. Jointly, these suggest that using adversarially filtered or perturbed datasets naively for benchmarking models is problematic.

We find that adversarially *collected* datasets ANLI (Nie et al., 2020a) and AdversarialQA (Bartolo et al., 2020) are also more challenging for all models while also showing signs of disproportionately disadvantaging the adversary model. However, with only a small number of such datasets available, it is difficult to draw strong conclusions about the overall efficacy or potential drawbacks of the approach. Importantly, unlike for adversarial filtering and perturbation, we cannot easily swap out the adversary model for analysis, as the adversarial data collection procedure can involve hundreds of hours of human labor per adversary.

In all three cases, our findings do not outright preclude the viability of adversarial dataset creation for evaluation purposes, but we urge researchers to keep these issues in mind when evaluating or comparing models based on adversarial datasets.

2 Related Work

AFLite is an adversarial filtering algorithm proposed by Sakaguchi et al. (2020), which also introduced Winogrande, an adversarial Winograd Schema Challange dataset. Le Bras et al. (2020) provided theoretical and empirical justification for AFLite, showing that models trained on AFLite-filtered data generalize better to out-of-domain datasets. Other datasets constructed using adversarial filtering include SWAG (Zellers et al., 2018) and HellaSwag (Zellers et al., 2019), two adversarially filtered commonsense multiple-choice datasets.

Given the over-parameterization of deep neural networks, adversarial perturbations (Goodfellow et al., 2015) have been identified as a particular weakness of these models. Within NLP, most adversarial attacks tend to occur at the token or word rather than continuous embedding level. TextFooler (Jin et al., 2019) is a popular adversarial textual perturbation method that swaps out words for synonyms while attempting to retain the semantic content and grammaticality of the original example. TextBugger (Li et al., 2019) works in a similar manner by searching for replacement words in the neighborhood within a context-aware embedding space, as well as using character-level edits. BERT-ATTACK (Li et al., 2020) and BAE (Garg and Ramakrishnan, 2020) use BERT's mask language modeling capability to substitute tokens.

An alternative approach is to collect data using a model in the loop, where human example-writers are given immediate feedback on whether an adversary model is able to correctly answer their example, and are incentivized to write examples on which the models fail. ANLI (Nie et al., 2020b) is an adversarial NLI dataset with multiple rounds of data collection. Williams et al. (2020) further provide fine-grained analysis of examples arising from ANLI's creation procedure. Bartolo et al. (2020) introduce AdversarialQA, an adversarial question-answering dataset. Kiela et al. (2021) further extend this approach, building a platform for continuous human-and-model-in-the-loop data creation. Using adversarially collected data as training data has been shown to lead to better performance on other adversarial datasets, but worse on out-of-domain datasets (Kaushik et al., 2021; Bowman et al., 2020). However, models trained on adversari-

ally collected data through many successive rounds have been shown to attain better performance (Wallace et al., 2021). In this work, we choose instead to focus exclusively on using adversarial examples as evaluation data.

In concurrent work, Adversarial Glue (Wang et al., 2021) applying a range of textual adversarial attacks to a subset of GLUE tasks to build a new language understanding benchmark. Importantly, they find that many adversarial attacks are prone to generating invalid examples, and perform careful manual filtering of the resulting examples.

3 Experimental Setup

Models The focus of our investigation is how the filtered dataset changes based on the choice of the adversary model. We consider a diverse set of pre-trained Transformer models: BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), XLM-R (Conneau et al., 2020), ELECTRA (Clark et al., 2020), MiniBERTa (Zhang et al., 2021), BART (Lewis et al., 2020), and DeBERTa-V2 and DeBERTa-V3 (He et al., 2021).

Tasks We consider four task datasets for our AFLite experiments. MNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015), Cosmos QA (Huang et al., 2019) and SocialIQA (Sap et al., 2019). We chose these tasks based on several criteria: having a large enough training set to be suitable for AFLite, being in a format suitable for AFLite (i.e. classification), and no model-adversarial procedure already having been applied in the creation of the dataset. We use MNLI and SNLI for TextFooler experiments, as TextFooler was designed with NLI tasks in mind.

Fine-Tuning For all models, we execute two separate fine-tuning setups. First, we perform full fine-tuning on the training set, across 3 random restarts. Second, we perform fine-tuning on a smaller held-out subset of training examples[1]–these will serve as the representations $\Phi(X)$ for AFLite, as well as the adversary models for TextFooler. We also repeat this subsampling across 3 random seeds, performing fine-tuning for each one. All of the AFLite and TextFooler results are averaged across the 3 fine-tuning and 3 AFLite runs. Refer to Appendix E for more details. All models were trained using `jiant` (Phang et al., 2020), which is built on Transformers (Wolf et al., 2020) and PyTorch (Paszke et al., 2019).

Model	MNLI	SNLI	Cosmos	SIQA
MiniBERTa-S-1M	60.2	73.4	41.6	42.4
MiniBERTa-B-1B	79.3	87.2	55.0	57.3
BERT-Base	82.7	89.5	57.8	59.8
XLM-R-Base	81.2	87.4	59.3	63.1
BART-Base	84.6	89.8	63.4	65.2
BERT-Large	85.5	91.0	61.9	65.5
ALBERT-Large	86.3	89.9	62.3	68.5
RoBERTa-Base	86.1	91.1	67.1	69.6
ALBERT-XLarge	87.2	91.6	70.9	71.2
XLM-R-Large	88.3	90.8	70.6	72.5
ELECTRA-Base	87.4	91.5	69.9	73.4
BART-Large	89.1	91.2	76.7	77.3
DeBERTa-V3-Base	89.8	92.6	74.4	77.7
RoBERTa-Large	89.6	91.8	78.5	77.4
ELECTRA-Large	90.3	92.7	83.2	79.7
DeBERTa-V2-Large	90.5	92.7	85.5	79.1
DeBERTa-V2-XLarge	90.2	92.7	87.0	78.1
DeBERTa-V3-Large	90.8	93.1	87.6	81.2

Table 1: Accuracy (%) of fully fine-tuned models on full validation sets. Models are sorted in order of average performance across all four tasks.

Table 1 shows the performance of fully fine-tuned models on the validation set of each task. In this and subsequent visualizations, we sort the models based on the average full fine-tuned performance on the four tasks, from weakest to strongest.

4 Adversarially Filtering Evaluation Sets

AFLite (Sakaguchi et al., 2020; Le Bras et al., 2020) is an adversarial filtering algorithm that iteratively removes "easy" examples from a dataset. To apply AFLite, given a dataset $D = (X, Y)$ of inputs X and labels Y, we first compute a learned representation $\Phi(x)$ for each example based on the adversary model. In each iteration, we sample multiple random subsets of the remaining data, fit weak classifiers on the subsets and compute predictions on the held-out examples. If an example is predicted correctly by more than a threshold τ of weak classifiers, it is removed from the dataset. This procedure is repeated until the number of examples removed in an iteration falls below a set threshold, resulting in a reduced dataset. Details can be found in the original work (Le Bras et al., 2020).

Sakaguchi et al. (2020) and Le Bras et al. (2020) apply AFLite before applying train/validation/test splits. However, because we are interested in the impact of the adversarial filtering on evaluation datasets,[2] we do not want to use evaluation exam-

[1]This follows the AFLite protocol of training a set of weaker classifiers to learn representations that effectively serve as the adversary models.

[2]In our experiments, we use the validation set of each task as the evaluation set.

ples to train the weak classifiers or influence the filtering procedure. Hence, we tweak the AFLite algorithm to separately filter out evaluation examples. We accomplish this by running the standard AFLite on the training examples, but in each round, we use the same weak classifiers and removal criteria to filter out "easy" evaluation examples. We show our modified AFlite in Algorithm 1 in the Appendix.

4.1 Results on AFLite Across Adversary and Fine-tuned Models

We apply AFLite using all 18 models as $\Phi(X)$, and evaluate against each fully fine-tuned model. The statistics of the examples filtered per model and task can be found in Section B in the Appendix. Figure 1 shows the results of fine-tuned models on validation sets filtered via AFLite using different adversary models.[3]

Overall, using AFLite with stronger adversary models leads to lower performance across all fine-tuned models, across all four tasks. Using a sufficiently strong adversary model for filtering pushes the performance of all tuned models to only slightly above chance: For instance, while most models score 80-90% on the unfiltered MNLI validation set, filtering using AFLite with DeBERTa-V3-Large results in no model scoring better than 45%.

We also observe a mild pattern of the weakest models performing slightly better as stronger adversaries are used in MNLI, SNLI, and SocialIQA. One explanation is that weaker models rely on easily learned heuristics (McCoy et al., 2019), and the weak classifiers in AFLite select examples that go against these heuristics, which weaker models subsequently perform poorly on. In contrast, stronger adversaries may filter out these examples.

4.1.1 Impact on Model Comparison

Evaluation datasets are often used to compare models, so we analyze the impact of adversarial filtering on the resulting sorting order of model performance. For each adversary model, we evaluate the fine-tuned models on the AFLite filtered dataset and sort the models by performance. In Figure 2, we show the ranked performance of models using different adversaries for a subset of the largest adversary models. We find that the sorting order of models is generally not consistent across adversary models. This is the case even if we ignore cases where the fine-tuned and adversary models share the same pretrained model.For MNLI and SNLI, evaluating on the datasets filtered by stronger adversaries appears to greatly distort the relative ranking of models. For Cosmos QA and SocialIQA, we observe that even when filtering with stronger adversaries, stronger models still tend to rank better than weaker models, but the ranking order is still not consistent across adversaries.

One interpretation of this result is that adversarial filtering may not give us evaluation data that is reliable for benchmarking and comparing models. An alternative interpretation is that as stronger adversary models are used, a larger proportion of remaining examples are challenging and therefore models are more likely to perform at chance on them. As such, we ought to expect stronger adversaries will lead to more randomness in the model rankings. In the extreme, if the weak classifiers in AFLite are as capable as the best-performing model, all models should perform at chance on the remaining examples. While performance on the strongest adversarially filtered datasets is still above chance for most models, we see that in MNLI and SNLI, all models converge to a small range of performance (35%–45%), meaning that a small variation in the number of correctly predicted examples can lead to a large change in model rank. This can lead to a distorted ranking of models.

We might also be concerned that the impact of adversarial filtering if the fine-tuned and adversary models are based on the same pretrained model. To measure this, we compute the rank of each model when no filtering is applied, and show how much the rank changes when filtering using the same pretrained model. However, as we show in Figure 10, the impact of filtering with the same pretrained model is disproportionately large, with all models except the weakest ones—which by definition cannot fall in rank—falling several positions in relative rankings. This implies that adversarial filtering for evaluation sets is very sensitive to the choice of model, and the resulting dataset is unfairly challenging if the adversary and evaluated models are based on the same pretrained model.

4.2 Label Agreement

To investigate the kinds of examples being identified as challenging via AFLite, we use the per-annotator labels of the MNLI and SNLI datasets. In the original data creation procedure, each validation-set example is annotated by 5 crowd-

[3] We present the same information in heatmaps in Figure 7 in the Appendix.

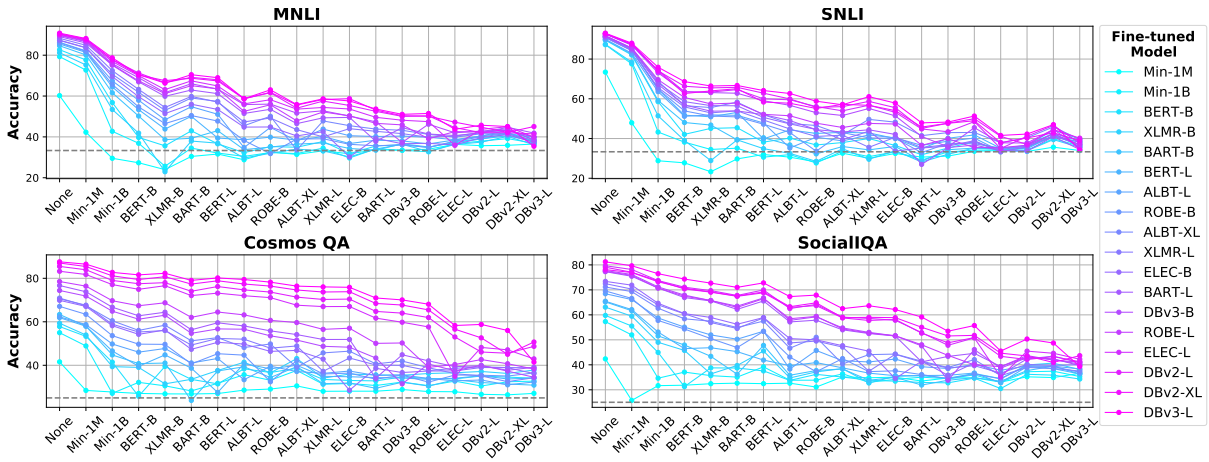


Figure 1: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. 'None' indicates the full unfiltered validation set. The dotted line indicates performance at chance for each task. Filtering with stronger adversary models leads to lower performance on the filtered dataset.

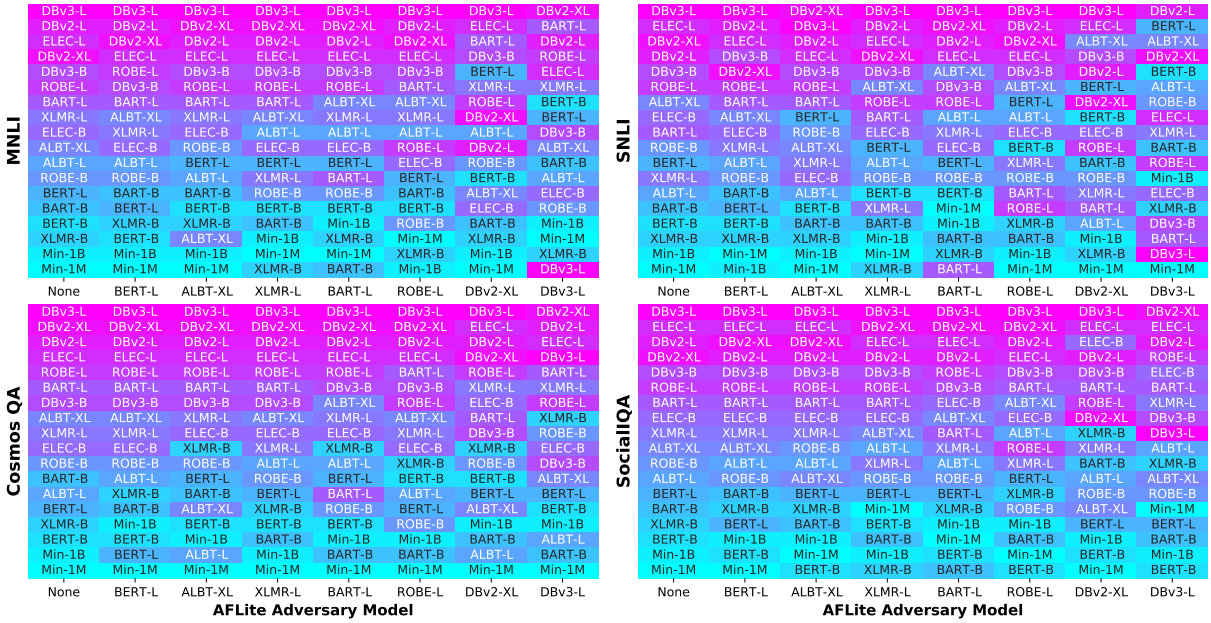


Figure 2: Ranked performance of fine-tuned models on validation sets filtered via AFLite using adversary models. We depict here a subset of the larger adversary models. For each AF Selected dataset, we sort models by their performance (Figure 1) from best (top) to worst (bottom). 'None' indicates the full validation set with no filtering applied. We find that the sorting order of model performance is not consistent across adversary models.

workers, and candidate examples are only accepted if at least 3 out of 5 crowdworkers agree on the label. We show in Figure 3 the average annotator agreement in the AFLite-selected examples across adversary models. For comparison, we also show the agreement rate among examples eliminated in the very first round of the AFLite procedure.

We observe a clear pattern across both datasets that filtering with stronger adversary models selects for examples with lower annotator agreement. Combined with our results above on lower model performance on filtered datasets, we take this as good evidence that the AFLite procedure indeed selects for the most challenging examples. It is unclear if these examples are challenging because they are genuinely difficult, where humans can easily make mistakes on them, genuinely ambiguous, or simply mislabeled. Conversely, we see that the first-pass filtered examples have consistently high annotator agreement, and that this rate does not vary across strength of the adversary models.

Oversampling low-agreement examples is not

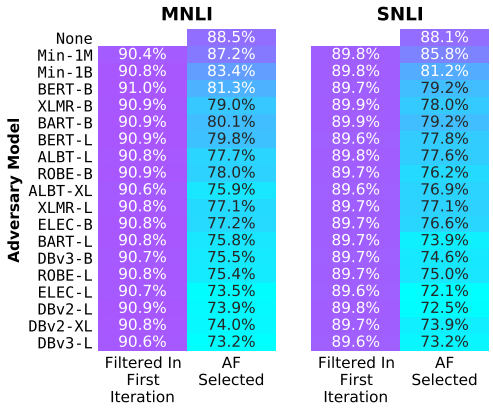


Figure 3: Label agreement among the adversarially filtered datasets from human annotators. *AF Selected* indicates examples that are not filtered out. *None* indicates no filtering applied i.e. agreement over the full validation set. Label agreement for the AF-selected datasets falls as better adversary models are used, indicating that AFLite may be selecting for the examples with the most ambiguity or labeling noise.

necessarily a bad thing if evaluated appropriately. Pavlick and Kwiatkowski (2019) and Nie et al. (2020c) show that there can be genuine disagreement between annotators over the example label, and argue that we should go beyond optimizing for model accuracy and instead train models to predict the full distribution of human judgements. The current format of scoring models on simple accuracy inadequate to evaluate on low-agreement examples, as the distribution of labels is reduced to a single label based on majority vote. Hence, if AFLite selects for low-agreement examples, one potential improvement would be to adjust the evaluation format to reflect annotator disagreement over labels.

5 Adversarially Perturbed Datasets

An alternative approach to creating more challenging datasets is to modify examples to lower the performance of a given adversary model. TextFooler (Jin et al., 2019) is a popular algorithm for adversarially perturbing examples, which involves swapping out words in the input for synonyms, with additional constraints aimed at retaining grammaticality and semantic content of the original example. We refer to the original work for full details of the TextFooler algorithm.

We apply the TextFooler methods to the SNLI and MNLI datasets, following the setup of the original work, where only the hypotheses are perturbed. For full comparability of results, we use the same models for $\Phi(X)$ in AFLite as the adversaries for TextFooler, and evaluate using the models fine-tuned on the full training set. We also use the same hyperparameters for the TextFooler algorithm as in the original work.

Overall, we find that adversarial perturbations impact model evaluation in a similar manner to adversarial filtering. TextFooler adversarial perturbations generally lower model performance across the board, with stronger adversaries leading to lower average performance across all models. However, we also observe a strong trend of models performing much worse on data perturbed adversarially to the same pretrained model, as seen by the distinct dip in performance along the diagonals of Figure 12. Moreover, using stronger adversaries also distorts the order of model performance, as seen in Figure 4, much moreso than with adversarial filtering. For instance, using DeBERTa-V3-Large as the adversary, ALBERT-XLarge is the best performing model; however, with ALBERT-XLarge as the adversary, DeBERTa-V3-Large is in turn the best performing model. The inconsistent impact on model performance across adversary models makes the naive usage of adversarial perturbations to create more challenging evaluation datasets problematic.

One major concern is whether the TextFooler-perturbed examples still retain the same semantic content and, more importantly, the labels of the original examples. Jin et al. evaluated a 100 examples from the validation set and found that the perturbations somewhat distorted the grammaticality and the meaning of the SNLI examples with a BERT adversary. We also observe a fall in quality of the examples in our experiments, with label flips arising from modifying key words, which greatly hurts the validity of performance obtained on these examples. We show a number of examples perturbed with different adversary models in (Table 4). Without any additional checks, using automatically perturbed examples for evaluating models can lead to extremely misleading results. Works such as AdversarialGLUE (Wang et al., 2021) perform an additional step of human validation and filtering of model-perturbed examples, which are necessary to ensure reliable evaluation data.

6 Model-in-the-Loop Adversarially Collected Datasets

In model-in-the-loop adversarial data collection, human crowdworkers are tasked with writing examples that a given adversary model will incorrectly

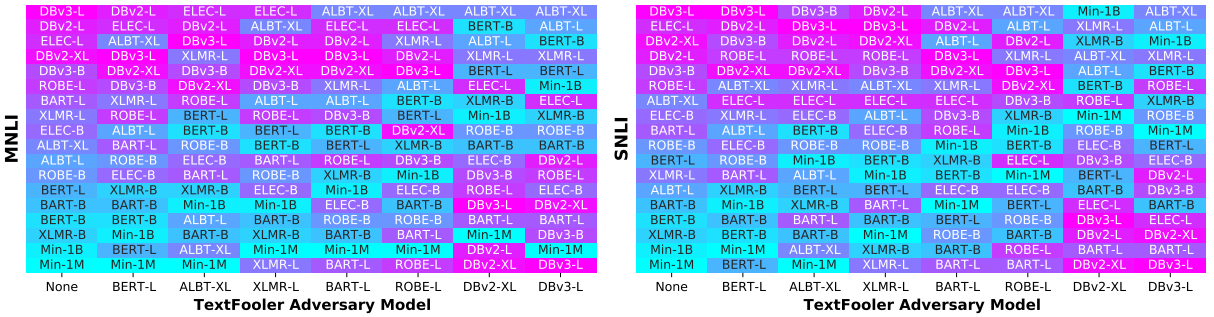
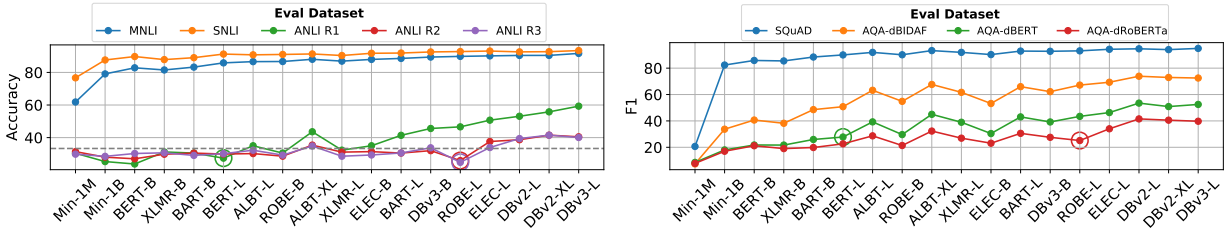


Figure 4: Ranked performance of fine-tuned models on validation sets perturbed via TextFooler using adversary models. For each perturbed dataset, we sort models by their performance (Figure 1) from best (top) to worst (bottom). 'None' indicates the full validation set with no filtering applied. We find that the sorting order of model performance is not consistent across adversary models, with dramatic reversals in many cases.



(a) ANLI (b) AdversarialQA

Figure 5: Measuring the performance of models on adversarially collected datasets. Exact Match scores for AdversarialQA are shown in Figure 13 in the Appendix. For each adversarially created dataset, the corresponding base adversary model used in model-in-the-loop data creation is circled in the corresponding color for that dataset. Performance at chance on ANLI is shown with a dotted line. While adversarial dataset creation appears to create datasets that are slightly harder for the adversary model compared to other models, the resulting datasets are harder across the board for all models, with stronger models still performing relatively better.

label. We consider two established model-in-the-loop adversarially collected datasets. ANLI (Nie et al., 2020b) is an NLI dataset adversarially collected through three iterative rounds, where the data for each round is written to be adversarial to models trained on a combination of MNLI, SNLI, and data from previous rounds. BERT-Large is used as the adversary model for round 1 of data collection, while RoBERTA-Large is used for rounds 2 and 3. AdversarialQA (Bartolo et al., 2020), is an adversarial question-answering dataset in the format of SQuAD 1.1 (Rajpurkar et al., 2016). Unlike ANLI, it consists of separately collected examples based on three adversary models: BiDAF (Seo et al., 2017), BERT-Large, and RoBERTa-Large.

While both datasets come with training, validation and test data splits, we conduct our analysis on the validation data. For both datasets, we fine-tune models on the conventional training data for each task,[4] before evaluating on both the standard and adversarial validation datasets.

[4]MNLI and SNLI for ANLI, and SQuAD 1.1 for AdversarialQA.

Figure 5 shows results on both model-in-the-loop datasets. For ANLI, about half of the models perform at chance for ANLI R1, whereas the stronger models perform significantly above chance. On the other hand, for ANLI R2 and R3, most models perform at chance except for the largest DeBERTa models. These results show that the ANLI data-generating procedure leads to examples that are harder for all models. However, we also observe that for ANLI R2 and R3, the performance of the adversary model, RoBERTa-large, is markedly below chance. This supports our observation above that while adversarial dataset creation can lower performance across the board, it still tends to hurt the adversary model more than others.

We see similar results for AdversarialQA, with models performing poorer as the datasets are generated with stronger adversaries. Unlike for ANLI, models do significantly better than chance on the adversarial datasets, with almost all models staying above 20% F1 and 10% EM.

Compared to our more extensive experiments on adversarial filtering, there are fewer datasets

collected using different adversary models, given the financial cost and manual writing needed to obtain examples. Hence we cannot draw strong conclusions about the efficacy of adversarial data collection for evaluation data from the current set of results. Moreover, the adversaries used in ANLI and AdversarialQA are not among the strongest models we used in our adversarial filtering experiments, where we saw the greatest distortion in the ranking of models. However, we do find that adversarial data collection leads to harder examples with stronger adversary models. As more work is done on adversarially collecting datasets and building benchmarks based on them (Kiela et al., 2021), we recommend that researchers pay close attention to the impact of the choice of adversary model and evaluate across a range of different models.

7 Discussion

We highlight that this work has not investigated the nature of the adversarial examples outside of the impact on model performance and annotator agreement. Works such as Williams et al. (2020) will be important for understanding exactly what examples are considered adversarial and why they are challenging to different models.

While our adversarial filtering experiments were performed on single adversary models, a possible alternative is to ensemble a diverse set of adversary models when running AFLite, or weight examples based on the AFLite example selection based on each adversary. This approach may help reduce the issue of disproportionate impact on any given adversary model's performance, and weighting evaluation across different example subsets may also potentially reduce the unstable ranking of models. However, this would significantly increase the cost of running the algorithm, would not address the issue of oversampling low-agreement examples, and may simply create a bias in favor of models that are greatly dissimilar from the pool of adversaries (e.g. non-Transformer models).

8 Conclusion

In this work, we have investigated three different approaches to adversarially constructing more challenging evaluation datasets int extensive experiments across 18 different pretrained models.

While all three methods result in lower scores as stronger adversaries are used, our takeaways on the viability of adversarial data creation to construct more challenging evaluation datasets are mixed.

Using a modified AFLite to adversarially filter evaluation examples, we find that there is a disproportionately large impact on the performance of fine-tuned models derived from the same pretrained model as the adversary, that the resulting ranking of models is unstable across the choice of adversary model especially as stronger adversaries are used, and that the filtering selects for examples with low annotator agreement over labels. On the other hand, the impact on model rankings is somewhat expected as a higher proportion of difficult examples remain after filtering.

Using TextFooler to perturb examples, we find even greater distortion in model rankings with stronger adversaries, and that examples can often be perturbed to the point of flipping labels, which is dire for model evaluation.

In a smaller set of experiments on adversarially collected datasets, we find hints of datasets being more challenging for the same pretrained model as the adversary, consistent with (Nie et al., 2020a), but are unable to draw stronger conclusions given the small number of adversaries used.

As the cost of using models goes down and their capabilities improve, we are likely to see more attempts to involve them in dataset creation. Models may be used adversarially as discussed above, or used to assist in writing examples via text generation models, or used in others ways, such as automatically identifying outliers or low-quality human-written examples. In any of these cases, it is possible to create an adverse and undesirable feedback loop in the data creation procedure.

While we believe that adversarially data methods can helpful in to creating more challenging evaluation benchmarks, we should take extra care to avoid the pitfalls of these approaches. Careful human validation of examples, as in (Wang et al., 2021), can help to address some pitfalls of adversarial approaches, but addressing biases at the dataset rather than example level can still be challenging. Importantly, adversarial datasets must accurately reflect the core task or capability being measured, ideally with a diverse set of examples that have good coverage of the linguistic phenomena associated with the task. As it stands, we do not yet find any free lunch for creating more challenging evaluation datasets.

References

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. *Transactions of the Association for Computational Linguistics*, 8:662–678.

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 (EMNLP)*. Association for Computational Linguistics.

Samuel R. Bowman and George Dahl. 2021. What will it take to fix benchmarking in natural language understanding? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4843–4855, Online. Association for Computational Linguistics.

Samuel R. Bowman, Jennimaria Palomaki, Livio Baldini Soares, and Emily Pitler. 2020. New protocols and negative results for textual entailment data collection. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8203–8214, Online. Association for Computational Linguistics.

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

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *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.

Siddhant Garg and Goutham Ramakrishnan. 2020. BAE: BERT-based adversarial examples for text classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6174–6181, Online. Association for Computational Linguistics.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2015. Explaining and harnessing adversarial examples.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. 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 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2019. Is bert really robust? natural language attack on text classification and entailment. *arXiv preprint arXiv:1907.11932*.

Divyansh Kaushik, Douwe Kiela, Zachary C. Lipton, and Wen-tau Yih. 2021. On the efficacy of adversarial data collection for question answering: Results from a large-scale randomized study. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6618–6633, Online. Association for Computational Linguistics.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. *arXiv preprint*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020*.

Ronan Le Bras, Swabha Swayamdipta, Chandra Bhagavatula, Rowan Zellers, Matthew Peters, Ashish Sabharwal, and Yejin Choi. 2020. Adversarial filters of dataset biases. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1078–1088. PMLR.

Gyeongbok Lee, Seung-won Hwang, and Hyunsouk Cho. 2020. SQuAD2-CR: Semi-supervised annotation for cause and rationales for unanswerability in SQuAD 2.0. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5425–5432, Marseille, France. European Language Resources Association.

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.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. *Proceedings 2019 Network and Distributed System Security Symposium*.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint*.

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.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020a. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020b. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Yixin Nie, Xiang Zhou, and Mohit Bansal. 2020c. What can we learn from collective human opinions on natural language inference data? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9131–9143, Online. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d' Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Ellie Pavlick and Tom Kwiatkowski. 2019. Inherent disagreements in human textual inferences. *Transactions of the Association for Computational Linguistics*, 7:677–694.

Jason Phang, Phil Yeres, Jesse Swanson, Haokun Liu, Ian F. Tenney, Phu Mon Htut, Clara Vania, Alex Wang, and Samuel R. Bowman. 2020. `jiant` 2.0: A software toolkit for research on general-purpose text understanding models. `http://jiant.info/`.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. 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 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Eric Wallace, Adina Williams, Robin Jia, and Douwe Kiela. 2021. Analyzing dynamic adversarial training data in the limit.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 3266–3280. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Boxin Wang, Chejian Xu, Shuohang Wang, Zhe Gan, Yu Cheng, Jianfeng Gao, Ahmed Hassan Awadallah, and Bo Li. 2021. Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models. *CoRR*, abs/2111.02840.

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. Association for Computational Linguistics.

Adina Williams, Tristan Thrush, and Douwe Kiela. 2020. ANLIzing the adversarial natural language inference dataset. *arXiv preprint*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Yian Zhang, Alex Warstadt, Xiaocheng Li, and Samuel R. Bowman. 2021. When do you need billions of words of pretraining data? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1112–1125, Online. Association for Computational Linguistics.

Algorithm 1: AFLite for Evaluation Data

Input: training dataset $D_T = (X_T, Y_T)$, evaluation dataset $D_V = (X_V, Y_V)$, pre-computed representation $(\Phi(X_T), \Phi(X_V))$, model family $\mathcal{M}$, target dataset size n, number of random partitions m, training set size $t < n$, slice size $k \leq n$, early-stopping threshold τ

Output: Filtering history of evaluation examples H, remaining evaluation examples R

$S = D_T$
$R = D_V$
while $|S| > n$ **do**
  // Filtering phase
  forall $i \in S$ **do**
    Initialize multiset of out-of-sample training predictions $E_T(i)$;
  forall $i \in R$ **do**
    Initialize multiset of out-of-sample evaluation predictions $E_V(i)$;
  for iteration *j : 1..m* **do**
    Randomly partition S into $(T_j, S \setminus T_j)$ s.t. $|S \setminus T_j| = t$;
    Train a classifier $\mathcal{L} \in \mathcal{M}$ on $\{(\Phi(x), y) | (x, y) \in S \setminus T_j\}$;
    forall $i = (x, y) \in T_j$ **do**
      Add the prediction $\mathcal{L}(\Phi(x))$ to $E_T(i)$;
    forall $i = (x, y) \in R$ **do**
      Add the prediction $\mathcal{L}(\Phi(x))$ to $E_V(i)$;
  forall $i = (x, y) \in S$ **do**
    Compute the predictability score $\tilde{p}(i) = |\{\hat{y} \in E_T(i) \text{ s.t. } \hat{y} = y\}| / |E_T(i)|$;
  forall $i = (x, y) \in R$ **do**
    Compute the predictability score $\tilde{p}(i) = |\{\hat{y} \in E_V(i) \text{ s.t. } \hat{y} = y\}| / |E_V(i)|$;
  Select up to k instances S' in S with the highest predictability scores subject to $\tilde{p}(i) \geq \tau$;
  $S = S \setminus S'$;
  Select all instances R' in R where $\tilde{p}(i) \geq \tau$;
  $R = R \setminus R'$;
  Append R' to H;
  if |S'| < k **then**
    break;
return H, R

A Modified AFLite

Algorithm 1, shows the modified AFLite algorithm, where the original algorithm applied to training examples is shown in black, and the additional lines applied to the evaluation examples are highlighted in red.

$\Phi(X)$ is the `CLS` or <S> embeddings of corresponding adversary model, fine-tuned on a separate held-out training set for the task (10% of the training data, following AFLte).

B AFLite Filtering Statistics

Figure 6 shows the breakdown of filtered examples when applying AFLite with different models. Each example in the validation set falls into one of three categories: examples filtered out on the first iteration of AFLite, examples filtered in all subsequent iterations, and examples remaining after applying AFLite (AF Selected). In most cases, more than half the validation datasets are filtered out within the first iteration, meaning that these examples were largely correctly predicted by a set of weak classifiers using the learned representations of partially tuned adversary models. Moreover, the stronger the adversary model, the more examples tend to be removed in the first iteration. Subsequent filtering iterations remove comparatively much fewer examples.

Among the AF Selected examples for Cosmos QA and SocialIQA, we see a trend that the stronger the adversary model, the fewer examples remain after AFLite. We do not see the same pattern in MNLI and SNLI, where the number of AF Selected examples does not vary consistently across strength of models. We note that Cosmos QA and SocialIQA use different AFLite hyperparameters from MNLI and SNLI because of the difference in datasets sizes (Table 3.

C Additional Results

Figure 8 shows the same information as Figure 1, with fine-tuned models on the X-axis and adversary models shown in different curves. Figure 7 shows the same information in a heatmap. Figure 11 shows the average agreement across adversarially filtered datasets, including the agreement among subsequent iterations of AFLite. Figure 13 shows exact-match scores on the AdversarialQA datasets.

D Models

Table 2 shows additional details for each of the pretrained models used in our experiments.

E Fine-Tuning Details

For full fine-tuning, we fine-tune for 3 epochs for MNLI and SNLI, and 5 epochs for Cosmos QA and SocialIQA. For fine-tuning weak classifiers for $\Phi(x)$, we subsample 10% of the training examples for MNLI and SNLI, and 5000 examples for Cosmos QA and SocialIQA, fixing the subsamples across all models. We repeat the subsampling procedure three times. In both fine-tuning setups, we hold out 500 examples from the training set for early stopping. These training examples are held out for both full fine-tuning as well as the AFLite

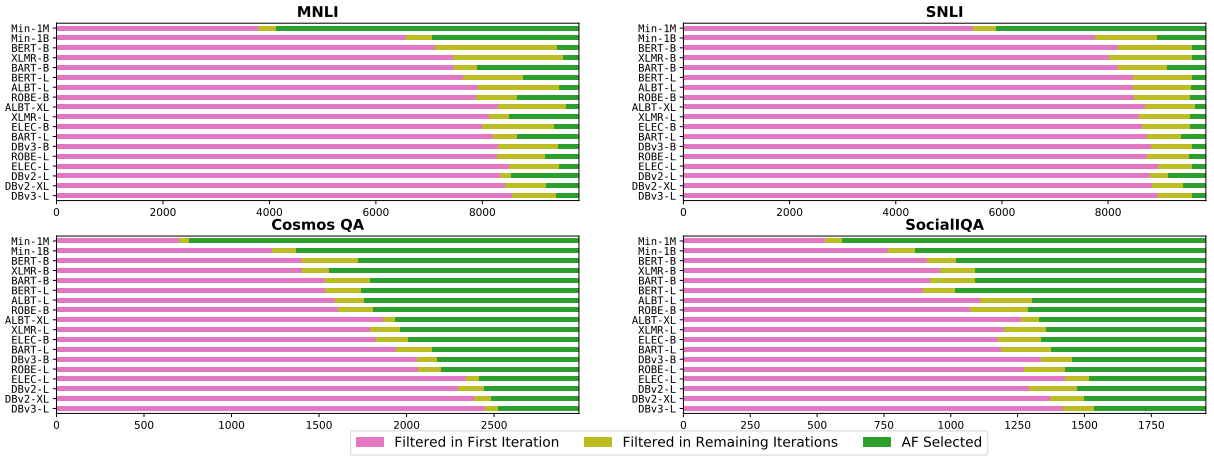


Figure 6: Statistics of AFLite-filtered datasets. We apply Algorithm 1 to the validation set of each task across adversary models, and average across three random seeds. *AF Selected* indicates examples that are included in the final evaluation set. For most models, majority of the examples are filtered out within the first iteration of AFLite.

Model	Abbreviation	Reference	Parameters	Training Objective
MiniBERTa Small 1M	Min-1M	Zhang et al. (2021)	∼45M	Masked language modeling
MiniBERTa Base 1B	Min-1B	Zhang et al. (2021)	∼100M	Masked language modeling
BERT-Base (cased)	BERT-B	Devlin et al. (2019)	∼100M	Masked language modeling + NSP
BERT-Large (cased)	BERT-L	Devlin et al. (2019)	∼340M	Masked language modeling + NSP
XLM-R-Base	XLMR-B	Conneau et al. (2020)	∼100M	Masked language modeling
XLM-R-Large	XLMR-L	Conneau et al. (2020)	∼340M	Masked language modeling
BART-Base	BART-B	Lewis et al. (2020)	∼100M	Text infilling + Sentence permutation
BART-Large	BART-L	Lewis et al. (2020)	∼340M	Text infilling + Sentence permutation
ALBERT-Large (v2)	ALB-L	Lan et al. (2020)	∼18M	Masked language modeling + SOP
ALBERT-XLarge (v2)	ALB-XL	Lan et al. (2020)	∼60M	Masked language modeling + SOP
RoBERTa-Base	RoBE-B	Liu et al. (2019)	∼100M	Masked language modeling
RoBERTa-Large	RoBE-L	Liu et al. (2019)	∼340M	Masked language modeling
ELECTRA-Base	ELEC-B	Clark et al. (2020)	∼100M	Replaced token detection
ELECTRA-Large	ELEC-L	Clark et al. (2020)	∼340M	Replaced token detection
DeBERTa-V2-Large (v2)	DBv2-L	He et al. (2021)	∼900M	Masked language modeling
DeBERTa-V2-XLarge (v2)	DBv2-XL	He et al. (2021)	∼1.5B	Masked language modeling
DeBERTa-V3 Base	DBv3-B	He et al. (2021)	∼100M	Replaced token detection
DeBERTa-V3 Large	DBv3-L	He et al. (2021)	∼418M	Replaced token detection

Table 2: Pretrained models used in our experiments

procedure. As such, validation examples never influence the fine-tuning or AFLite procedures, only being used when we perform AFLite and filter our validation examples as described in Algorithm 1.

For DeBERTa, unlike in He et al. (2020), we do not apply SiFT during fine-tuning.

F AFLite Hyperparameters

Table 3 shows the hyperparameters for our AFLite runs.

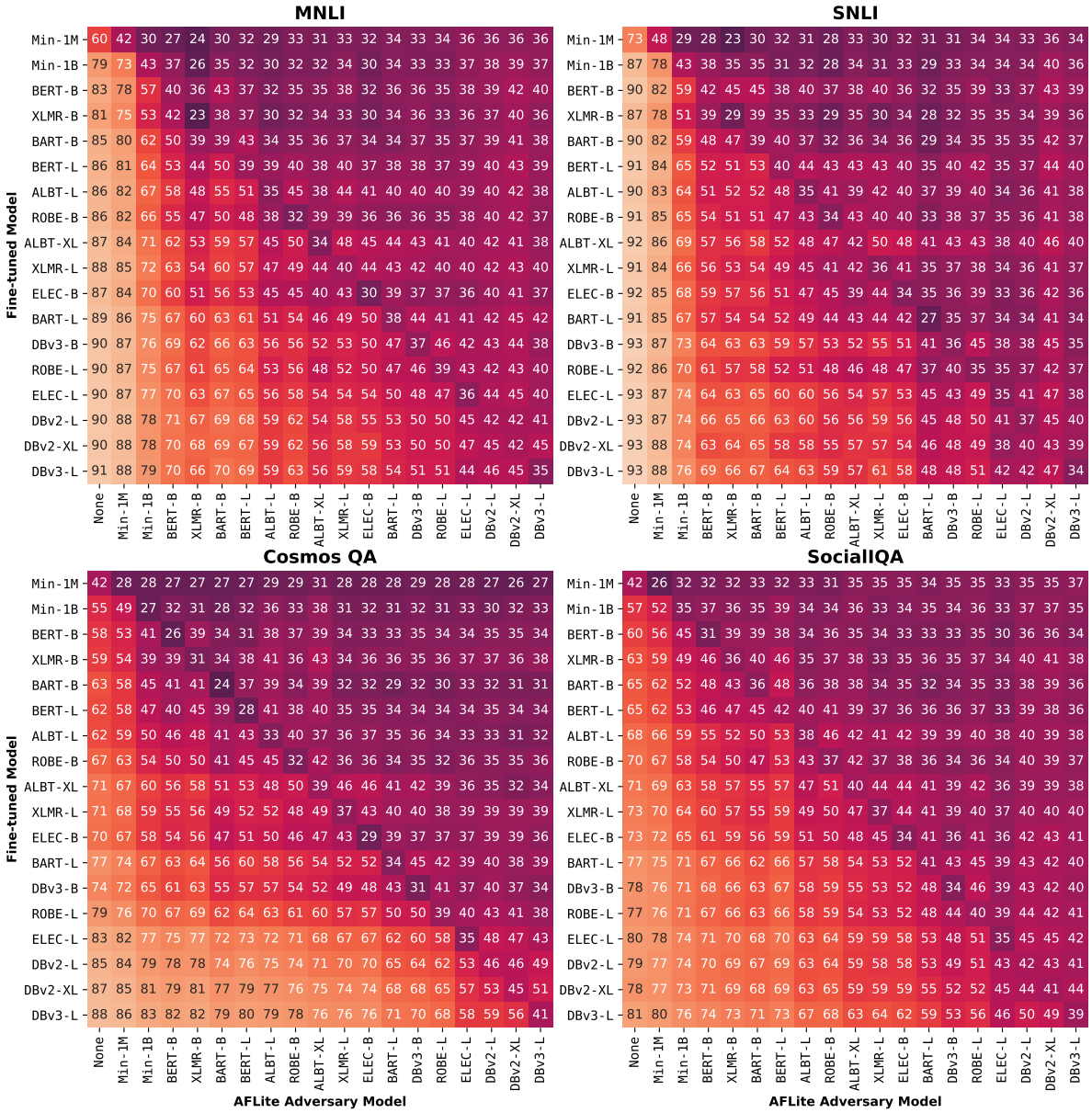


Figure 7: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. 'None' indicates the full validation set with no filtering applied. Filtering with stronger adversary models leads to lower performance on the filtered dataset, across all fine-tuned models. However, filtering also tends to hurt the adversary model itself more than other models on average (darker cells on the diagonal).

	MNLI	SNLI	Cosmos QA	SocialIQA
m	64	64	64	64
t	50K	40K	10k	10k
k	10K	10K	500	500
τ	0.75	0.75	0.75	0.75
Taken From	Le Bras et al. (2020)	Le Bras et al. (2020)	Sakaguchi et al. (2020)	Sakaguchi et al. (2020)

Table 3: AFLite Hyperparameters

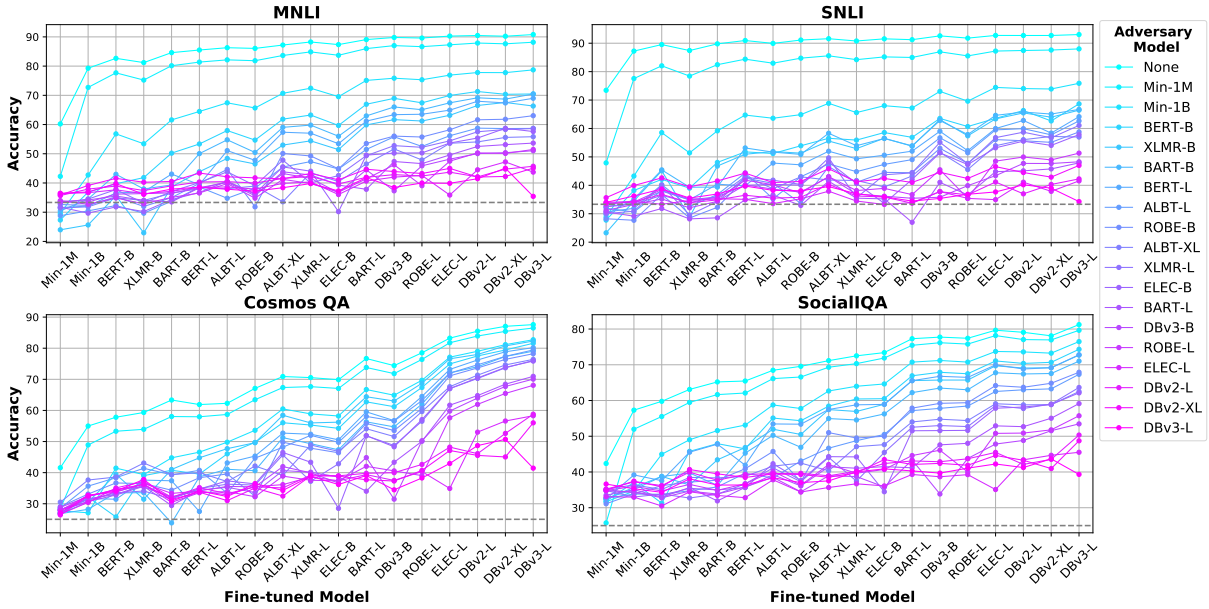


Figure 8: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. 'None' indicates the full validation set with no filtering applied. The dotted line indicates performance at chance for each task. Filtering with stronger adversary models leads to lower performance on the filtered dataset, across all fine-tuned models.

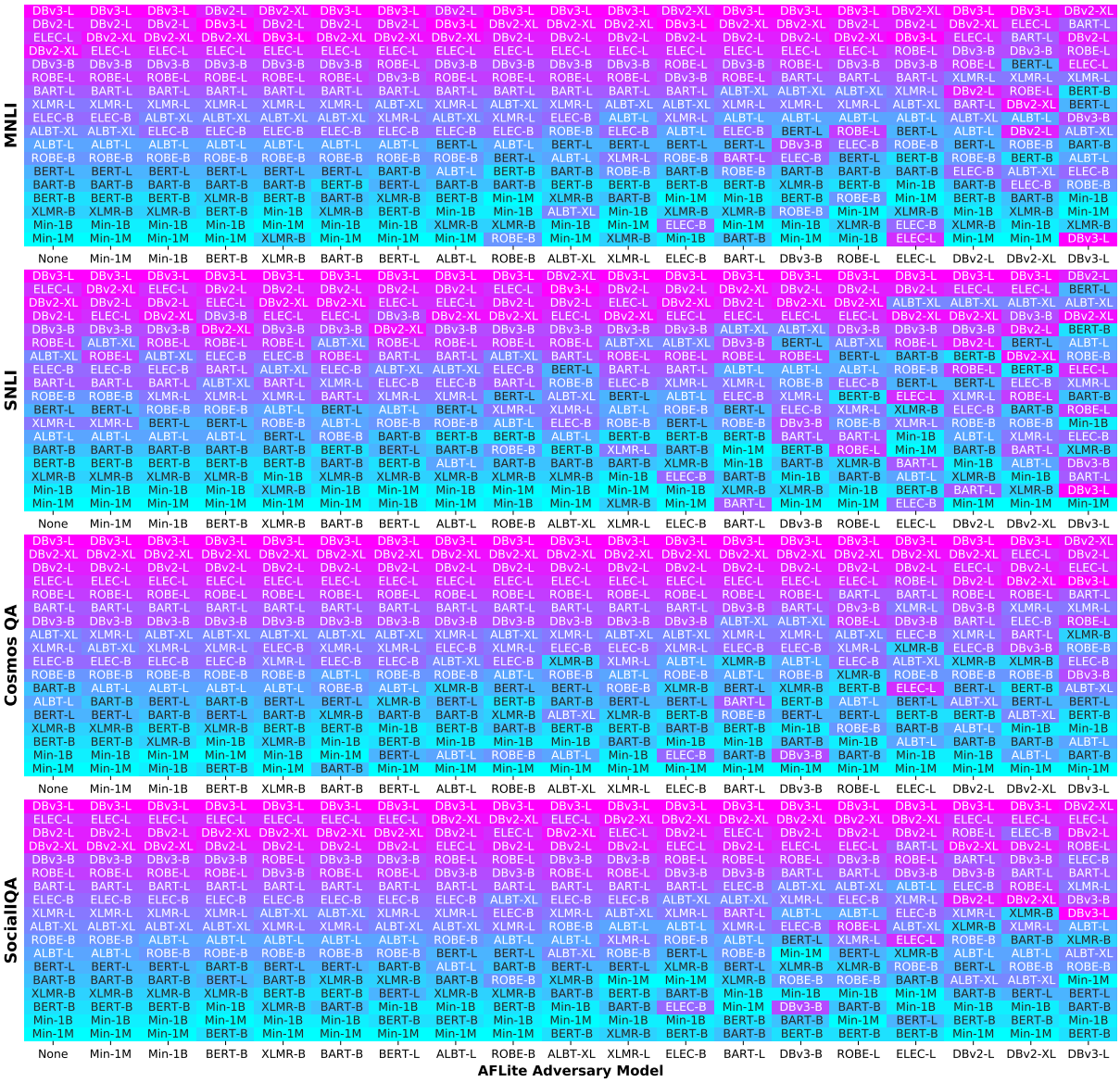


Figure 9: Ranked performance of fine-tuned models on validation sets filtered via AFLite using adversary models. We depict here a subset of the larger adversary models. For each AF Selected dataset, we sort models by their performance (Figure 1) from best (top) to worst (bottom). 'None' indicates the full validation set with no filtering applied. We find that the sorting order of model performance is not consistent across adversary models.

Premise	Original Hypothesis	Label	Perturbed Hypothesis	Issue
		Adversary: **BERT-Large**		
In Hong Kong you can have a plate, or even a whole dinner service, hand-painted to your own design.	It's impossible to have a plate hand-painted to your own design in Hong Kong.	contradiction	It's imaginable to have a plate hand-painted to your own design in Hong Kong.	Semantics change
He hadn't seen even pictures of such things since the few silent movies run in some of the little art theaters.	He had recently seen pictures depicting those things.	contradiction	He had recently experimented pictures exposing those things.	Problematic word substitution
As a basic guide, the symbols below have been used to indicate high-season rates in Hong Kong dollars, based on double occupancy, with bath or shower.	As you can see, the symbols are of dolphins and octopuses.	neutral	As you can see, the symbols are of sharks and octopuses.	Semantics change
		Adversary: **DeBERTa-V3-Large**		
Do you think Mrs. Inglethorp made a will leaving all her money to Miss Howard? I asked in a low voice, with some curiosity.	I yelled at the top of my lungs.	contradiction	I muttered at the superior of my lungs.	Problematic word substitution
But when the cushion is spent in a year or two, or when the next recession arrives, the disintermediating voters will find themselves playing the roles of budget analysts and tax wonks.	The cushion will likely be spent in under two years.	entailment	The cushion will likely be spent in under three years.	Semantics change
substitute my my yeah my kid'll do uh four or five hours this week for me no problem	I just can't make the time because of my job.	neutral	I just can't make the time for of my job.	Grammatical error

Table 4: TextFooler Examples

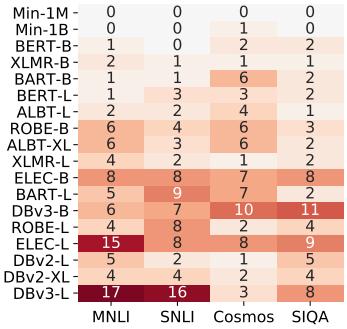


Figure 10: For each fine-tuned model, we compute the change in rank (1=best, 18=worst) from evaluating on the full evaluation set, and on the dataset filtered using the same pretrained model for the adversary. In almost all cases, filtering on the same pretrained model leads to a fall in ranking, indicating that the model is disproportionately affected by filtering with itself.

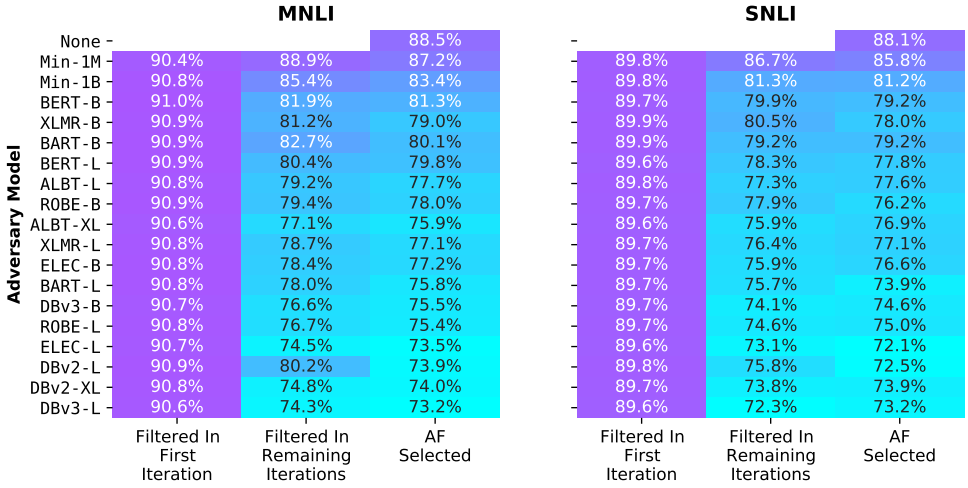


Figure 11: Label agreement among the adversarially filtered datasets from human annotators. *AF Selected* indicates examples that are not filtered out. Label agreement is very high for first pass filtered examples for all models. On the other hand, label agreement for the remainder datasets falls as better adversary models are used, indicating that AFLite may be selecting for the examples with the most ambiguity or labeling noise.

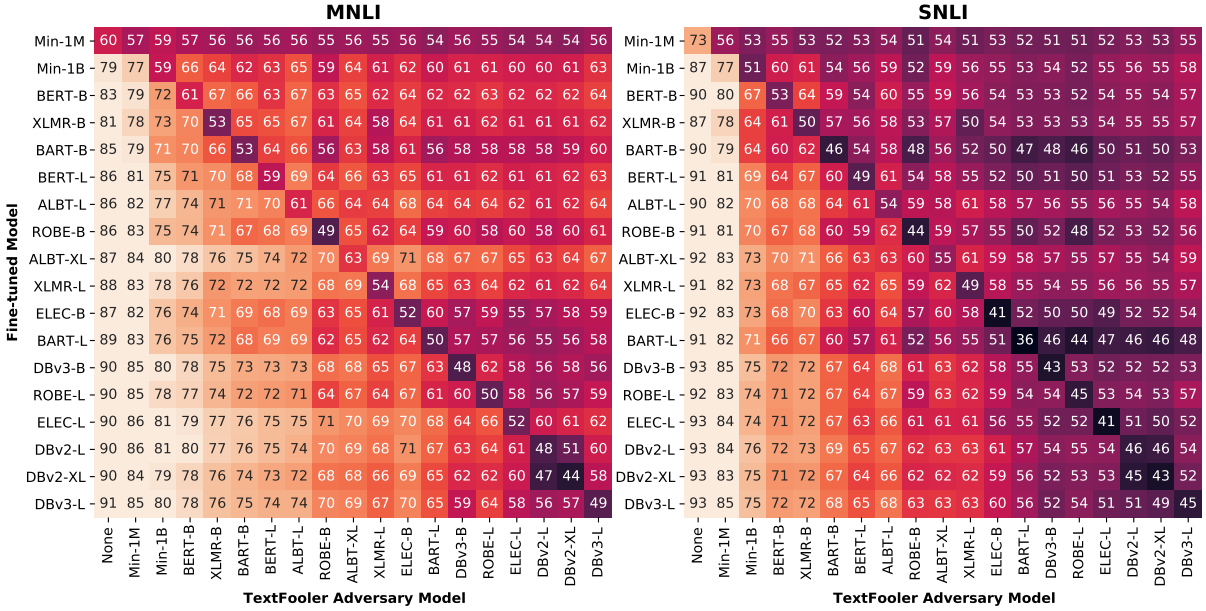


Figure 12: Performance of fine-tuned models on validation sets perturbed via TextFooler using adversary models. 'None' indicates the full validation set with no perturbation applied. Perturbing with stronger adversary models leads to lower performance on the filtered dataset, across all fine-tuned models. However, perturbing also tends to hurt the adversary model itself more than other models on average (darker cells on the diagonal).

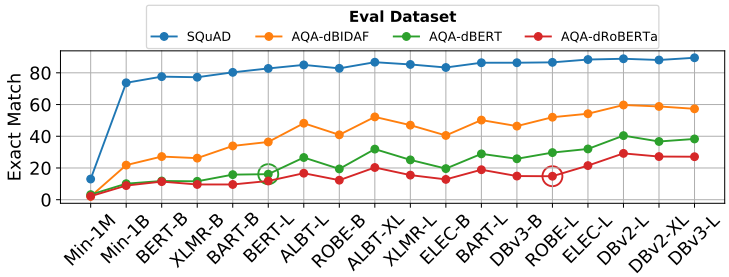


Figure 13: Measuring the performance of models on AdversarialQA. AdversarialQA models are fine-tuned on SQuAD 1.1. For each adversarially created dataset, the corresponding base adversary model used in model-in-the-loop data creation is circled in the corresponding color for that dataset.