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

Anonymous ACL submission

Abstract

More capable language models increasingly saturate existing task benchmarks, in some cases outperforming humans, leaving little headroom with which to measure further progress. Adversarial dataset creation has been proposed as a strategy to construct more challenging datasets, and two common approaches are: (1) filtering out easy examples and (2) model-in-the-loop data collection. In this work, we study the impact of applying each approach to create more challenging evaluation datasets. We adapt the AFLite algorithm to filter evaluation data, and run experiments against 18 different adversary models. We find that AFLite indeed selects more challenging examples, lowering the performance of evaluated models more as stronger adversary models are used. However, the resulting ranking of models can also be unstable and highly sensitive to the choice of adversary model used. Moreover, AFLite oversamples examples with low annotator agreement, meaning that model comparisons hinge on the most contentiously labeled examples. Smaller-scale experiments on the adversarially *collected* datasets ANLI and AdversarialQA show similar findings, broadly lowering performance with stronger adversaries while disproportionately affecting the adversary model.

1 Introduction

Large-scale language models have attained strong performance across a variety of language understanding tasks, including question-answering, natural language inference (NLI), and paraphrase identification. As the capabilities of these models improve, it has become increasingly difficult to systematically evaluate and benchmark further model improvements (Vania et al., 2021). 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 human baseline scores. This has left little headroom with which to measure further improvements in models and progress in NLP. More than ever, we need new approaches to build challenging and reliable evaluation datasets at scale (Bowman and Dahl, 2021).

Prior work such as Le Bras et al. (2020) and Nie et al. (2020a) have proposed adversarially filtering or constructing examples to raise the difficulty of task datasets, leveraging highly capable models 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. In the extreme, adversarial datasets may be so narrowly optimized toward stumping a particular model that they no longer accurately measure the abilities that the dataset was designed to test.

In contrast to prior 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 two 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, and (2) 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 a task dataset. We apply AFLite in extensive experiments across four English-language NLP datasets and 18 different models to study the interaction between the choice of adversary model and the resulting evaluation performance. 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 adversarial filtering and adversarial dataset collection do result in more challenging evaluation datasets, but they are not without their drawbacks. We find that the general outcome of adversarial *filtering* is to lower performance across the board, with stronger adversary models leading to more challenging subsets of examples. However, as more difficult evaluation subsets are identified, the relative order of model performance is not preserved, with large random variation in model ranks as stronger adversaries are used. This suggests that using adversarially filtered datasets for benchmarking models can be problematic. Performance on the filtered datasets is also much worse if the evaluated and adversary models are based on the same pretrained model, which can lead to the difficulty of the dataset being overstated. Adversarial filtering also oversamples examples with low annotator agreement, which could mean that these examples are contentious even for human annotators.

Similarly, 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.

In both cases, our findings do not 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

We perform most of our experiments using the AFLite 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 further theoretical and empirical justification for AFLite, showing that models train 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.

An alternative approach is to collect data using a model in the loop, where human example-writers are given immediate feedback on whether a trained adversary model is able to correctly answer their example, and are incentivized to write examples on which the models fail. Nie et al. (2020b) introduce ANLI, an adversarial NLI dataset with multiple rounds of data collection. Williams et al. (2020) provide fine-grained analysis of the kinds of examples arising from this adversarial dataset 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 adversarially collected data through many successive rounds have been shown to attain better performance (Wallace et al., 2021).

3 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. First, given a dataset $D = (X, Y)$ of inputs X and labels Y, we 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 data subsets and compute predictions on the held-out examples. If an example is predicted correctly by more than some 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. More details can be found in the original manuscript (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,[1] we do not want to use evaluation examples to train the weak classifiers or influence the

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

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. This modified procedure differs from the standard AFLite in two key ways: (1) There is no limit to how many evaluation examples can be removed in each round. Thus, it is common for many examples to be removed in the very first round of filtering. (2) Evaluation examples are not used in the fitting steps of the AFLite algorithm. We show our modified AFlite in Algorithm 1 in the Appendix.

4 Experimental Setup

Models The crux 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 and DeBERTa$_{\text{RTD}}$ (He et al., 2021).

Tasks We consider four task datasets for our experiments. MNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015) are natural language inference tasks, while Cosmos QA (Huang et al., 2019) and SocialiQA (Sap et al., 2019) are multiple-choice commonsense reasoning tasks. These tasks are chosen 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. All four tasks are scored with simple accuracy.

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, to supply the representations $\Phi(X)$ for AFLite, we perform fine-tuning on a held-out subset of training examples. We also repeat this subsampling across 3 random seeds, performing fine-tuning and AFLite for each one. All of our results on AFLite are averaged across the 3 fine-tuning and 3 AFLite runs. Refer to Appendix D 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$_{\text{RTD}}$-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-Large	90.5	92.7	85.5	79.1
DeBERTa-XLarge	90.2	92.7	87.0	78.1
DeBERTa$_{\text{RTD}}$-Large	90.8	93.1	87.6	81.2

Table 1: Performance (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.

5 AFLite on Evaluation Sets

5.1 AFLite Filtering Statistics

We show in Figure 1 the breakdown of 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

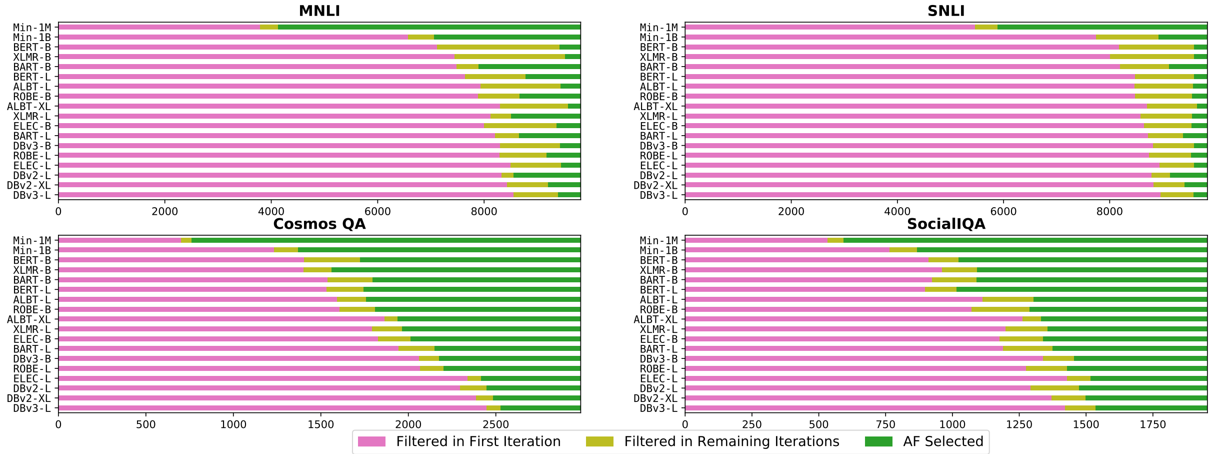


Figure 1: Statistics of AFLite-filtered datasets. We apply Algorithm 1 to the validation set of each task across adversary models, and average across three random seeds. *AF Selected* indicates examples that are not filtered out. For most models, majority of the examples are filtered out within the first iteration of AFLite.

datasets sizes (see Table 3 in the Appendix).

5.2 Results on AFLite Across Adversary and Fine-tuned Models

Figure 2 shows the results of fine-tuned models on validation sets filtered via AFLite using different adversary models.[2] We emphasize that the fine-tuned models that we evaluate are trained entirely separately from the partially tuned models used to learn representations $\Phi(X)$ used in AFLite.

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$_{\text{RTD}}$-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.

5.2.1 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 AF Selected dataset and sort the models by performance, as shown in Figure 3. 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, which we address below. 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

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

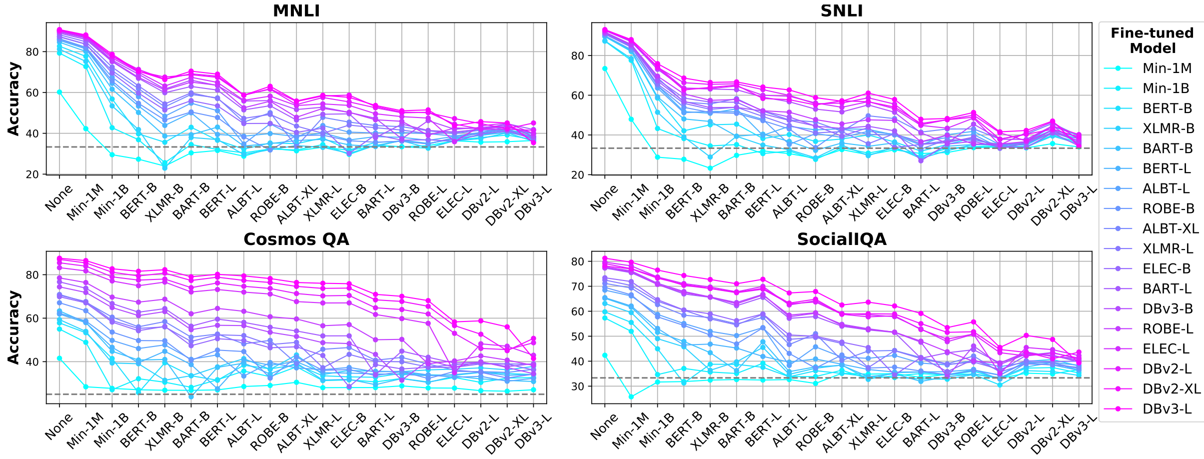


Figure 2: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. 'None' indicates the full unfiltered validation set. The dotted line indicates performance at chance for each task. Filtering with stronger adversary models leads to lower performance on the filtered dataset, across all fine-tuned models.

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 on performance might be disproportionately large 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. Ideally, if there is no model-specific bias to the filtering, there should be no change. However, as we show in Figure 4, 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 can be very sensitive to the choice of model, and the resulting dataset can be unfairly challenging if the adversary and evaluated models are based on the same pretrained model.

5.3 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 crowdworkers, and candidate examples are only accepted if at least 3 out of 5 crowdworkers agree on the label. We show in Figure 5 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 necessarily a bad thing if they are 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. As easy examples seem to be highly correlated with high annotator agreement, one potential approach to construct a more challenging and discriminative benchmark would be to identify low-agreement examples, acquire additional annotations, and train and evaluate models on predicting the distribution of human labels. However, the current format of scoring models on simple accuracy is an inadequate method of evaluating on low-agreement examples, as the distribution of labels is reduced to a single label based on majority vote. Hence, if AFLite

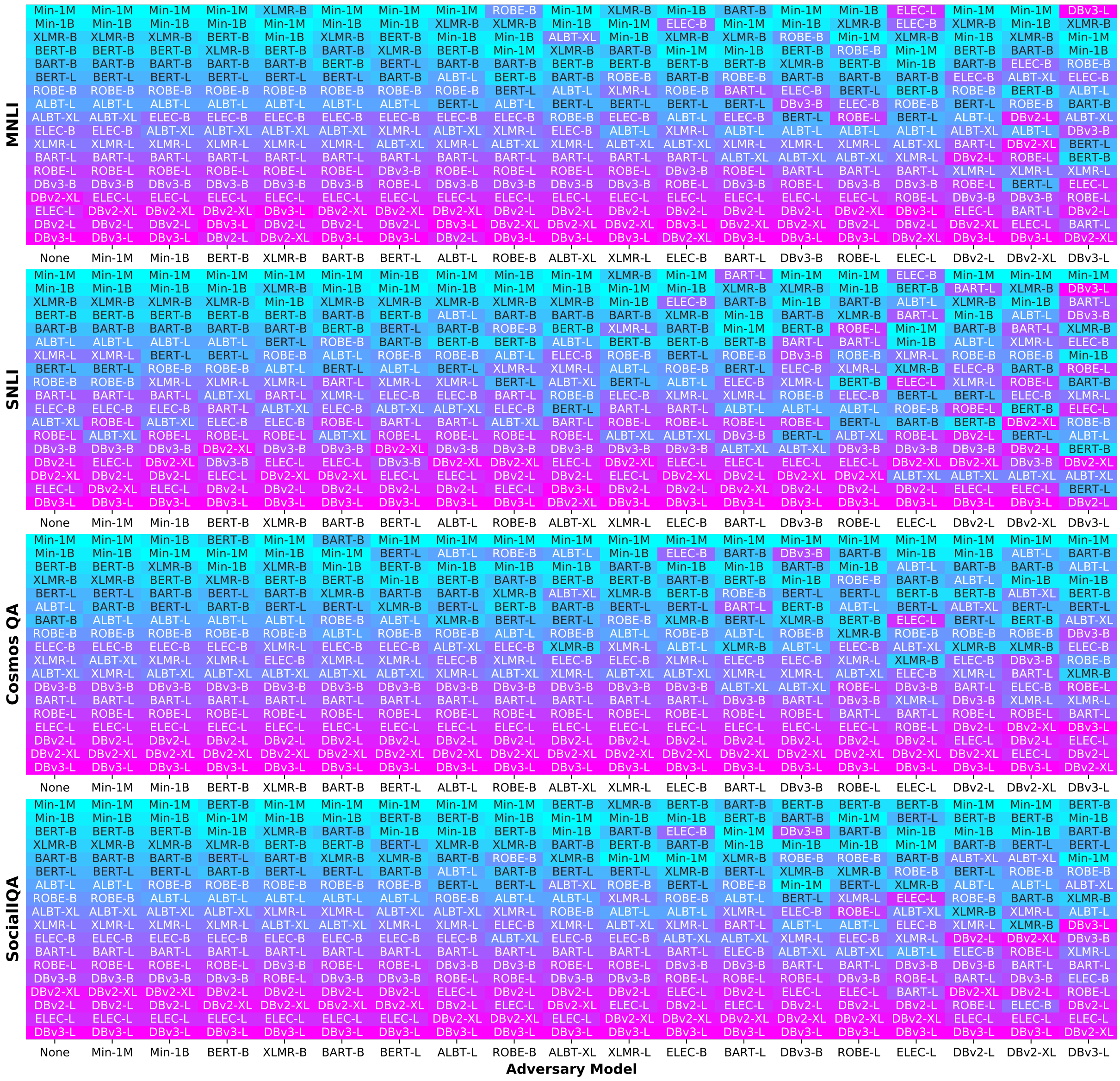


Figure 3: Ranked performance of fine-tuned models on validation sets filtered via AFLite using adversary models. For each AF Selected dataset, we sort models by their performance (Figure 2) from worst (top) to best (bottom). 'None' indicates the full validation set with no filtering applied. We find that the sorting order of model performance is not consistent across adversary models.

selects for low-agreement examples, the evaluation format should be adjusted according to accommodate the annotator disagreement over labels.

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 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

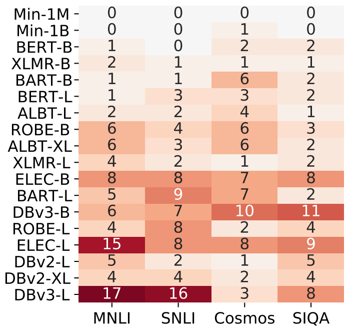


Figure 4: For each fine-tuned model, we compute the change in rank (1=best, 18=worst) from evaluating on the full evaluation set, and on the dataset filtered using the same pretrained model for the adversary. In almost all cases, filtering on the same pretrained model leads to a fall in ranking, indicating that the model is disproportionately affected by filtering with itself.

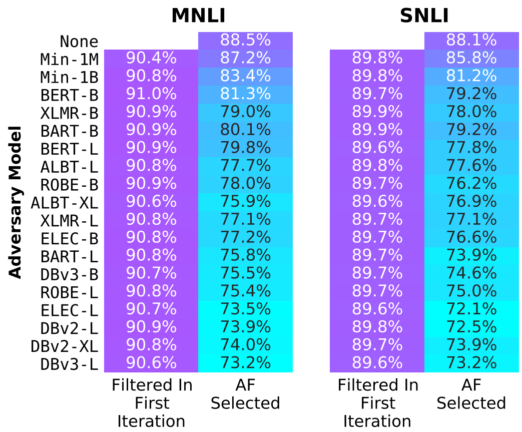


Figure 5: Label agreement among the adversarially filtered datasets from human annotators. *AF Selected* indicates examples that are not filtered out. *None* indicates no filtering applied i.e. agreement over the full validation set. Label agreement for the AF-selected datasets falls as better adversary models are used, indicating that AFLite may be selecting for the examples with the most ambiguity or labeling noise.

the validation data. For both datasets, we fine-tune models on the conventional training data for each task,[3] before evaluating on both the standard and adversarial validation datasets.

We show in Figure 6 results on both model-in-the-loop datasets. For each adversarially created dataset, we circle data points where the fine-tuned model is the same as the adversary model.For ANLI, we see that 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 more difficult across 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 obtaining above 20 F1 and 10 EM scores.

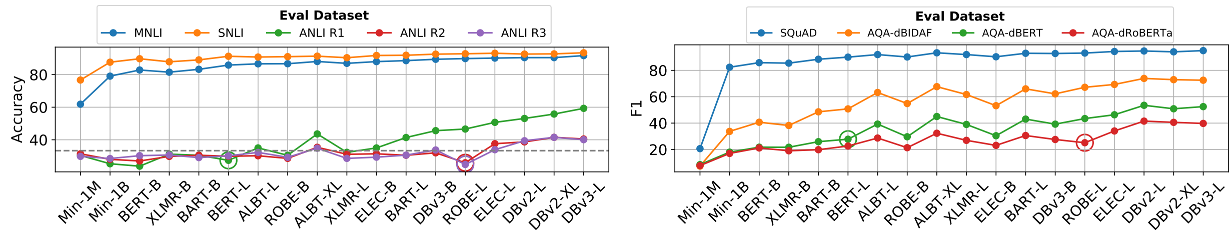
Compared to our more extensive experiments on adversarial filtering, there are fewer datasets collected using different adversary models, given the financial cost and manual writing needed to obtain examples. Hence we cannot draw strong conclusions about the efficacy of adversarial data collection for 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

One limitation of this study is that most of our models are encoder-only Transformer models, omitting sequence-to-sequence models such as T5 (Raffel et al., 2020) and GPT-3 (Brown et al., 2020), or non-Transformer models. However, our experiments do cover a diverse and comprehensive set of the prominently used models in the literature, many which have dominated benchmarking leaderboards, and spanning a wide range of sizes, pretraining objectives, and training corpora, making this still a highly relevant sample of models to study.

We also highlight that this work has not investigated the nature of the adversarial examples outside

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



(a) ANLI (b) AdversarialQA

Figure 6: Measuring the performance of models on adversarially collected datasets. Exact Match scores for AdversarialQA are shown in Figure 10 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.

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, and would not address the issue of oversampling low-agreement examples, which is consistent across all adversary models.

8 Conclusion

In this work, we have investigated two different approaches to adversarially constructing more challenging evaluation datasets.

Using a modified AFLite, we run extensive experiments performing adversarial filtering of evaluation examples and model evaluation across 18 different pretrained models. Our takeaways on the viability of adversarial filtering to create more challenging evaluation datasets are mixed. On one hand, there is a disproportionately large impact on the performance of fine-tuned models based on the same pretrained model as the adversary, the resulting ranking of models is unstable across the choice of adversary model, especially as stronger adversaries are used, and the filtering selects for examples with low annotator agreement over labels. On the other hand, the resulting datasets are indeed more challenging, the impact on model rankings is somewhat expected as a higher proportion of difficult examples remain after filtering, and low-agreement examples can be valuable *if* an appropriate evaluation format is used that takes into account the distribution of the labels.

On our smaller set of experiments on adversarially collected datasets, we draw a set of similar conclusions. Adversarial data collection leads to more challenging datasets, but there are signs of disproportionate impact on the adversary model.

As the cost of using models goes down and their capabilities improve, we are likely to see more involvement of models in dataset creation in the future. 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 constructing datasets can be a viable approach to create more challenging evaluation benchmarks, we should take extra care to avoid the pitfalls of these approaches. Importantly, adversarial datasets must still 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. For now, we recommend that researchers evaluate against a wide range of models where possible, and avoid measuring the difficulty of adversarial datasets using the adversary models themselves.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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.

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.

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.

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/`.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

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*.

Clara Vania, Phu Mon Htut, William Huang, Dhara Mungra, Richard Yuanzhe Pang, Jason Phang, Haokun Liu, Kyunghyun Cho, and Samuel R. Bowman. 2021. Comparing test sets with item response theory. 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 1141–1158, Online. Association for Computational Linguistics.

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.

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 $k \le n$, 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 $\tilde{p}(i) \ge \tau$;

  $S = S \setminus S'$;

  Select all instances R' in R where $\tilde{p}(i) \ge \tau$;

  $R = R \setminus R'$;

  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 Additional Results

Figure 8 shows the same information as Figure 2, 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 9 shows the average agreement across adversarially filtered datasets, including the agreement among subsequent iterations of AFLite. Figure 10 shows exact-match scores on the AdversarialQA datasets.

C Models

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

D 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 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.

E AFLite Hyperparameters

Table 3 shows the hyperparameters for our AFLite runs.

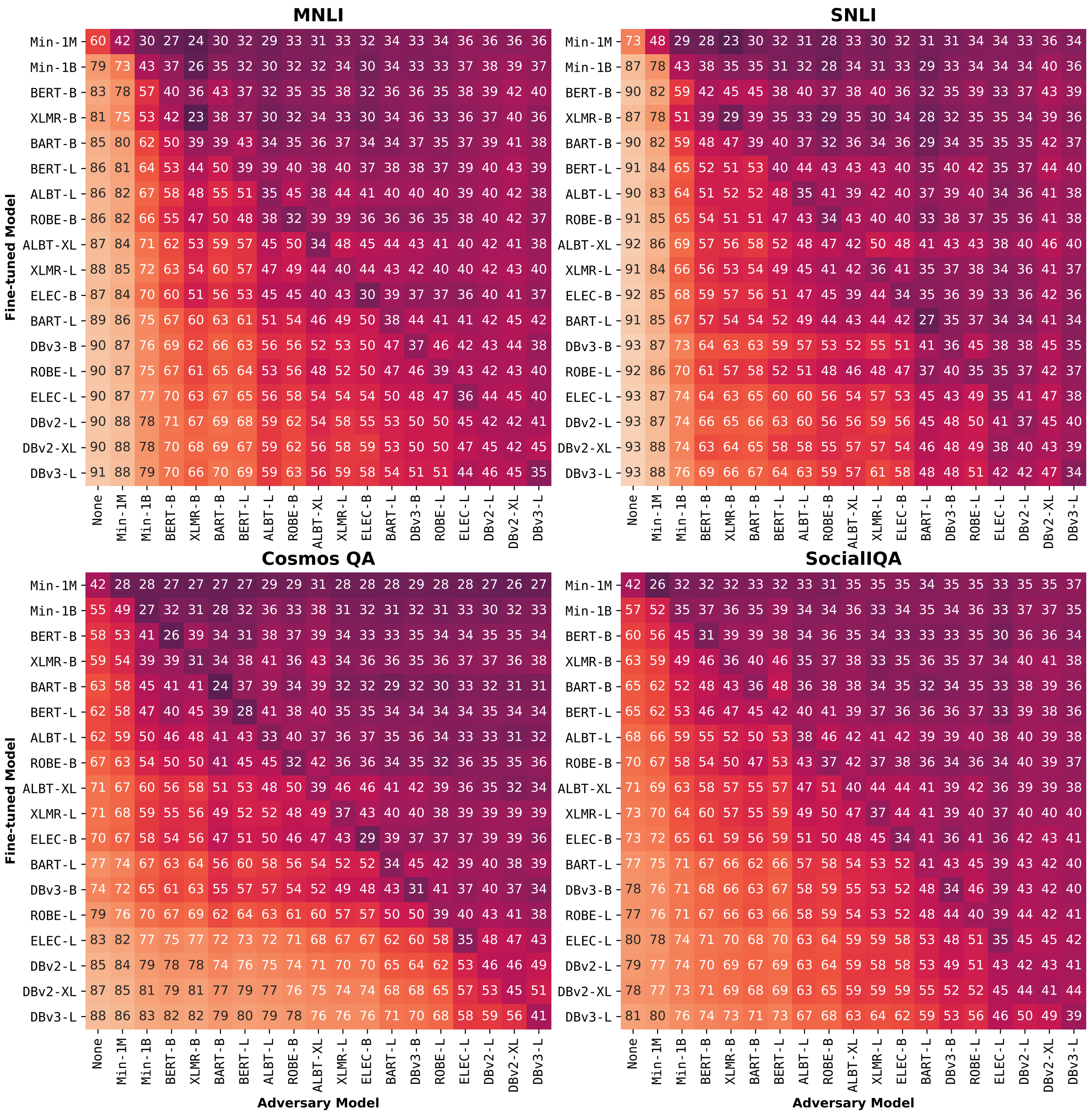


Figure 7: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. ‘None’ indicates the full validation set with no filtering applied. Filtering with stronger adversary models leads to lower performance on the filtered dataset, across all fine-tuned models. However,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

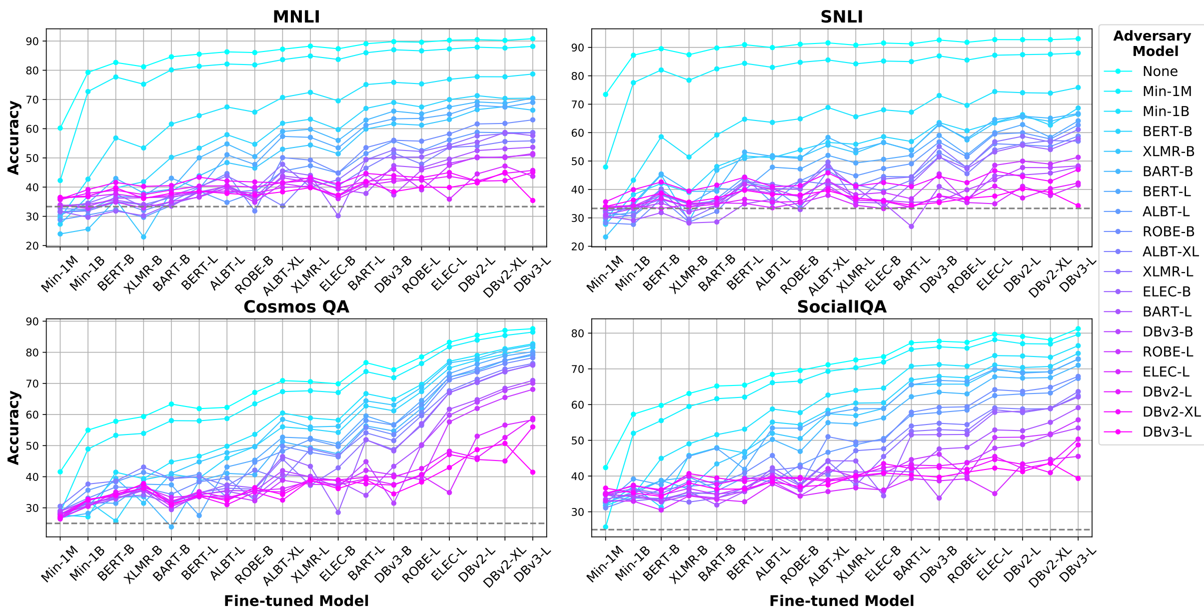


Figure 8: Performance of fine-tuned models on validation sets filtered via AFLite using adversary models. 'None' indicates the full validation set with no filtering applied. The dotted line indicates performance at chance for each task. Filtering with stronger adversary models leads to lower performance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

Adversary Model	MNLI Filtered In First Iteration	MNLI Filtered In Remaining Iterations	MNLI AF Selected	SNLI Filtered In First Iteration	SNLI Filtered In Remaining Iterations	SNLI AF Selected
None			88.5%			88.1%
Min-1M	90.4%	88.9%	87.2%	89.8%	86.7%	85.8%
Min-1B	90.8%	85.4%	83.4%	89.8%	81.3%	81.2%
BERT-B	91.0%	81.9%	81.3%	89.7%	79.9%	79.2%
XLMR-B	90.9%	81.2%	79.0%	89.9%	80.5%	78.0%
BART-B	90.9%	82.7%	80.1%	89.9%	79.2%	79.2%
BERT-L	90.9%	80.4%	79.8%	89.6%	78.3%	77.8%
ALBT-L	90.8%	79.2%	77.7%	89.8%	77.3%	77.6%
ROBE-B	90.9%	79.4%	78.0%	89.7%	77.9%	76.2%
ALBT-XL	90.6%	77.1%	75.9%	89.6%	75.9%	76.9%
XLMR-L	90.8%	78.7%	77.1%	89.7%	76.4%	77.1%
ELEC-B	90.8%	78.4%	77.2%	89.7%	75.9%	76.6%
BART-L	90.8%	78.0%	75.8%	89.7%	75.7%	73.9%
DBv3-B	90.7%	76.6%	75.5%	89.7%	74.1%	74.6%
ROBE-L	90.8%	76.7%	75.4%	89.7%	74.6%	75.0%
ELEC-L	90.7%	74.5%	73.5%	89.6%	73.1%	72.1%
DBv2-L	90.9%	80.2%	73.9%	89.8%	75.8%	72.5%
DBv2-XL	90.8%	74.8%	74.0%	89.7%	73.8%	73.9%
DBv3-L	90.6%	74.3%	73.2%	89.6%	72.3%	73.2%

Figure 9: 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.

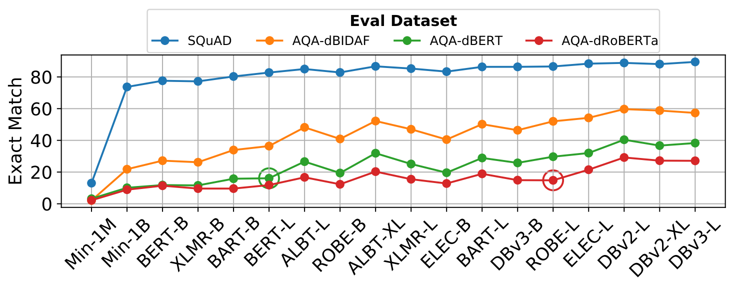


Figure 10: Measuring the performance of models on AdversarialQA. AdversarialQA models are fine-tuned on SQuAD 1.1. For each adversarially created dataset, the corresponding base adversary model used in model-in-the-loop data creation is circled in the corresponding color for that dataset.

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

Table 2: Pretrained models used in our experiments

	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