Analyzing Fine-Tuned ELECTRA Performance on Benchmark Datasets & Mitigating Answer Length Bias

Subhan Chaudry
schaudry@utexas.edu
University of Texas at Austin
Austin, Texas, USA

Abstract

Pretrained language models are typically able to perform very well on readily available benchmark datasets, but there are concerns as to whether they are truly learning reasoning or simply repeating patterns observed in the data. In this experiment, a single pretrained ELECTRA-small model was applied for finetuning against two benchmark datasets, the Stanford Natural Language Inference (SNLI) for Natural Language Inference and the Stanford Question Answering Dataset (SQuAD) for Question Answering. A shared training pipeline was designed to display strong performance initially.

In terms of baseline performance, SNLI achieved accuracy of 90% while SQuAD demonstrated 78.1% for EM (Exact Match) and 85.9% on F1. Through observing the SNLI predictions, it was clear that the model performs well except in that it confuses neutral with entailment frequently. For SQuAD, the model was shown to decrease in performance for long answer spans with exact matches dropping to 60.4% from 81.4% for gold answers with six or more tokens.

The project investigated the usage of oversampling of training examples to improve performance for longer answers. This approach was shown to improve F1 scores on long answers from 78.4% to 80.3%. However, this came with a tradeoff of overall performance of the model. The results show that there is potential to mitigate answer length bias and other specific improvements, but they may result in differences for model behavior overall.

1 Introduction

Pretrained transformer models are commonly being leveraged for many natural language processing (NLP) tasks, including both Natural Language Inference (NLI) and Question Answering (QA). Developers typically will fine-tune one of these pretrained models on individual tasks or benchmarking datasets and report analysis metrics like accuracy or F1 scores. However, a critical point to note is that even if models display similar scores on these metrics, they can still perform differently on similar queries or tasks, like handling negations differently or trading off performance on small vs longer answers. They may also simply be using inherent patterns in the training set as opposed to true language understanding.

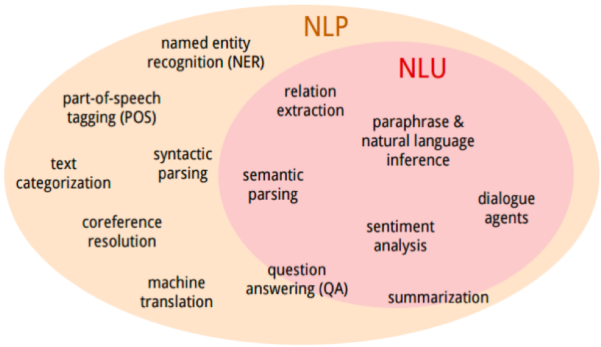

Figure 1. Natural Language Processing Tasks

In this project, the google/electra-small-discriminator model was observed to compare performance between NLI and QA benchmarking datasets and determine ways to improve overall measures of these models with different techniques. SNLI was used for NLI tasks, while SQuAD was taken for extractive QA tasks. Standard models were applied upon both of these datasets to build an understanding of behavior with a simple fine-tuning approach and identification of key weaknesses in training/implementation.

For the training pipeline, a training infrastructure was provided by Course Staff for Natural Language Processing at the University of Texas at Austin. This was based on Hugging Face Transformers being applied to fine-tune the ELECTRA-small model on SNLI and SQuAD datasets. A separate analysis script was developed to review the individual predictions generated and generate insights like confusion matrices, recall, performance on slices, and performance based on answer lengths. This allowed for identification of key gaps in the fine-tuned models to improve the training pipeline.

The initial fine-tuned models demonstrated strong performance, but one key weakness was observed in the lower EM and F1 scores for longer answers in the model trained on SQuAD. To improve the model, this project looked at leveraging oversampling to increase the number of long answers in the training set. The default model training and oversampling approach are then compared to understand

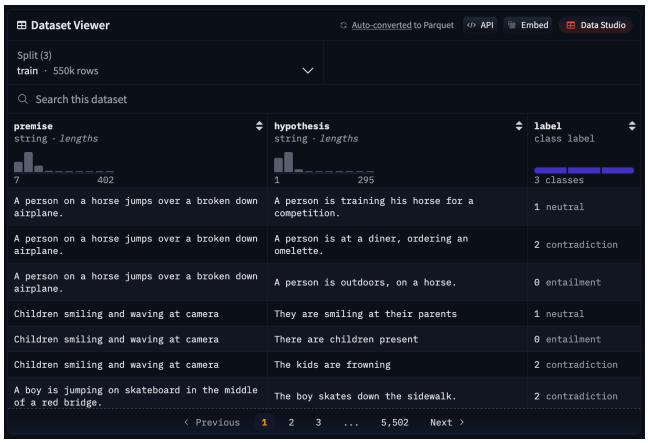

Figure 2. The Stanford Natural Language Inference (SNLI) Dataset Training Examples

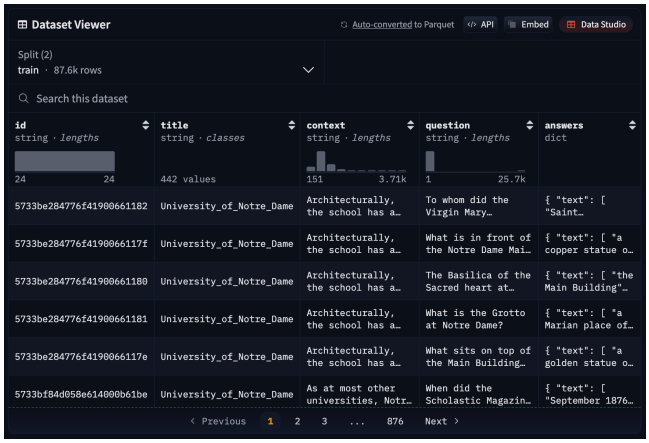

Figure 3. The Stanford Question Answering Dataset (SQuAD) Training Examples

the tradeoffs of focusing on mitigating answer length bias in the final model.

2 Related Work

Previous studies have been done to analyze the performance of language models trained with benchmark datasets, and in identifying potential ways to mitigate bias and improve performance. Poliak et al. [1] published research where hypothesis-only baselines were reaching around 69% accuracy on SNLI models.

When it comes to question answering, Kaushik and Lipton [2] looked at baselines for popular datasets and found that question and passage-only models were able to perform very well. They concluded that SQuAD and CNN were better constructed in general.

There are many approaches to reduce bias in language models like ensemble-based models focused on residuals [3], data augmentation [4], and contrastive training [5] methodologies. The work in this project simplifies many of these potential approaches by looking at oversampling to improve training without much refactoring of the the code provided.

3 Methodology

This project includes a standard workflow for data preparation, fine-tuned model training, model evaluation, model improvement, and analysis of improved models.

3.1 Datasets and Tasks

For the Natural Language Inference task, the Stanford Natural Language Inference (SNLI) dataset was applied. SNLI contains 570k human-written English sentence pairs labeled for classification with entailment, contradiction, and neutral labels. The model was provided the training split with evaluation taking place on the validation split with any examples with empty labels being filtered out.

For the Question Answering task, the Stanford Question Answering Dataset (SQuAD) was leveraged instead. SQuAD included questions from a set of Wikipedia articles with the answer being the corresponding reading passage span. The model is provided with the question and context and then trained to predict the answer span from the context. The SQuAD EM and F1 measures are used to evaluate the performance of the model.

3.2 Model and Training

The google/electra-small-discriminator model serves as the base model for fine-tuning against both of the baseline datasets. ELECTRA is an approach for self-supervised language representation learning that trains models to tell real input tokens from fakes ones from another network.

The provided run.py script creates the sequence classification head for the NLI task and the span prediction head for the QA task. The model is then fine-tuned on the relevant dataset for each task with tokenization.

Models ran through training for three epochs using the default hyperparameters. For each task, there is a distinct training and evaluation command applied with two generated outputs:

- eval_metrics.json which produces the metrics like accuracy, EM, and F1 scores.
- eval_predictions.jsonl that contains a JSON object for each of the examples from the validation dataset applied.

3.3 Analysis of Models

For further analysis of the model performance, an analysis.py script was drafted with the help of Anthropic's Claude Opus 4.5 and OpenAI's ChatGPT 5.1 models to pull out key statistics from the models. This script ingests the eval_predictions.jsonl

file for each of the models and then computes specific metrics for NLI and QA to help identify gaps in performance and areas for overall improvement.

3.4 QA Model Improvement

Following the generation of the analysis output to investigate model performance, the base training code provided by Course Staff was adapted to improve the performance and mitigate identified biases. As discussed later in the Part I: Analysis section, answer length was one area where the QA model seemed to perform worse on long answers. Oversampling code was introduced to the run.py script to improve the training dataset and then the QA model was reran with these improvements with analysis being generated once again.

4 Part I: Analysis

Once both models for NLI and QA completed training, they were analyzed to understand how the models were performing and identify specific weaknesses in their implementation. The analysis script along with the eval_predictions.jsonl files were used to look closely at overall model performance.

4.1 NLI Analysis

Looking at the Natural Language Inference task, the model performed fairly well with the default hyperparameters. The model evaluated with 9,842 examples in total and achieved an accuracy of 0.899 on the SNLI dataset. This is in line with the expected accuracy for the ELECTRA-small model trained on SNLI, showcasing that the benchmark datasets alone can be sufficient for training effective models.

To take a closer look at how errors were distributed, we can look at the confusion matrix below in Table 1 to see the gold and predicted labels for each of the different classification categories.

Table 1. NLI Confusion Matrix (rows = gold, cols = predicted)

	Entailment	**Neutral**	**Contradiction**
Entailment	3037	224	68
Neutral	224	2807	204
Contradiction	69	207	3002

We can observe that the model does not perform identically on each of the classes. Looking more closely at this matrix, the per-class recall values can be seen below. This gives a closer look as to whether the model was struggling with any specific class.

- Entailment: 3037 / 3329 ≈ 91.2%
- Neutral: 2807 / 3235 ≈ 86.8%
- Contradiction: 3002 / 3278 ≈ 91.6%

Interestingly, the neutral class has the highest amount of misclassifications with the lowest recall at 86.8%. This could be due to the fact that it is harder for the model to handle cases where the hypothesis is neither supported nor contradicted. Instead it seems to prefer a stronger prediction to either entailment or contradiction.

The analysis script generated five examples for each type of error pair which can allow for looking more closely at the types of errors commonly experienced by the model.

When we look specifically at contradiction to neutral errors, many are due to the model not having enough implicit reasoning to handle these specific scenarios. We can look at the example below, where the model predicts neutral since it does not understand that waiting at the amusement park and waiting to see a movie are not related to one another and cannot occur at the same time. It seems to simply equate both examples of 'waiting' with one another.

- Premise: "Families waiting in line at an amusement park"
- Hypothesis: "People are waiting to see a movie"
- Gold Label: Contradiction
- Predicted: Neutral

We can also look at entailment to neutral errors with another example below. In this scenario, the model seems to be unable to perform reasoning to understand that these are actually related to one another since there are no similar words between the sentences. It seems to act more conservatively in general when looking at additional errors as well.

- Premise: "People are throwing tomatoes at each other"
- Hypothesis: "The people are having a food fight"
- Gold Label: Entailment
- Predicted: Neutral

When we look specifically at negations, which are hypotheses containing 'not' or 'n't', the model continues to show strong performance at 90.4%. This is typically a challenge for NLP models and demonstrates that the model is fairly effective with just the benchmark dataset.

4.2 QA Analysis

For the Question Answering task, the model similarly had strong performance with just the default hyperparameters. However, we measure performance as a measure of the Exact Match (EM) and F1 scores. The model evaluated 10,570 examples in total and achieved an EM score of 78.1% and F1 score of 85.9% on SQuAD. This is to be expected for the ELECTRA-small model trained on this benchmark dataset as well.

For a deeper analysis, the next step was to look at the performance of the model on different answer lengths to see if there were differences in the scores. Table 2 breaks this down by short, medium, and long answers which were returned by the analysis script.

Based on this table, there are significant differences in the score based on the provided answer lengths. There is a drop in EM of 21.0% and a drop in F1 of 8.4% from short to long

Table 2. QA Performance by Answer Length

Answer Length	Count	EM	F1
Short (≤2 tokens)	6,654	81.4%	86.8%
Medium (3–5 tokens)	2,877	77.0%	86.7%
Long (≥6 tokens)	1,039	60.4%	78.4%

answers. This shows that the overall model performance is not equivalent across answer lengths.

This is likely due to the fact that the SQuAD dataset has mostly short answers with 63.0% of the examples having two or less tokens. On the other hand, long answers with six or more tokens only consist of 9.8% of the dataset. This can cause the model to become biased towards shorter spans when it makes predictions. The larger gap between EM and F1 scores also points to the fact that the spans predicted by the model tend to overlap with the gold label but do not match exactly as frequently as other answer lengths.

The analysis script also produced an EM of 74.0% and F1 of 81.7% for the negation examples, but this is not as significant as the differences in the answer lengths.

To summarize, after looking at examples of the QA task, it seems that the bias observed in the answer length could be due to the following reasons:

1. **Dataset Composition**: There is a higher number of short answers in the training dataset which can cause gradient updates that incentivize shorter span predictions rather than longer ones.
2. **Overlap Errors**: The QA task requires for models to return both the starting and ending positions. Longer answers can have trouble identifying the spans as they have larger margin for error due to more words.
3. **Complex Reasoning**: Answers that are longer rely on more information from context tokens which make it a more difficult task, especially when producing larger spans.

5 Part II: Fixing It

Following the analysis conducted in Part I, it was clear that the main gap of the models existed in the bias related to answer lengths. This is where Part II was spent to improve on the performance of long answers.

5.1 Oversampling Approach

The simplest approach to addressing the answer length biases was to leverage an oversampling approach to increase the number of long answer examples present in our training dataset. The goal was to introduce more of these to the model in the hopes that it would be become better at learning patterns within them.

The code was adapted with the following changes:

1. Examples with a normalized answer length beyond the 6+ token threshold were placed into a set.
2. These long answer examples were replicated based on the provided oversampling factor.
3. These examples were then combined with the training dataset to increase their representation.
4. The dataset is then shuffled to evenly distribute them across the training iterations.

5.2 Implementation in Training

To execute the oversampling approach within the project, two command line arguments were added to the training code:

- `-oversample_long_answers`: This argument helps improve QA by oversampling training examples where the gold answer length is ≥ this threshold
- `-oversample_factor`: This argument defines the replication factor for oversampling long-answer QA examples.

The rest of the code and training process remained the same as what was done for the benchmark datasets in Part I. The key advantage to this approach is that it allows for minimal architecture changes with the focus placed on the dataset distribution itself. There is no other modification of the dataset that could result in irregular behavior.

6 Results

6.1 Model Performance

Following the implementation of the changes in Part II, the analysis script was run once again to compare the model with oversampling against the original base model trained in the initial fine-tuning. Table 3 breaks down the performance differences between these models.

Table 3. Comparing Base vs Oversampling Overall

Metric	Base	Oversampled	Δ
Overall EM	78.1%	76.3%	−1.8%
Overall F1	85.9%	85.1%	−0.8%
Long EM	60.4%	60.7%	+0.3%
Long F1	78.4%	80.3%	+1.9%

These results show that there is a tradeoff that can be seen when trying to mitigate longer answer bias rather than overall model performance. The oversampling approach was successfully able to improve EM by 0.3% and F1 by 1.9% for the long answer examples. However, this model performed worse overall. We can take an even closer look with Table 4 and directly compare the different answer lengths and how well they perform for each of the models.

It becomes evident that the model is choosing to weigh more heavily towards long answers for improvement thereby

Table 4. Comparing Performance by Answer Length

Length	Base EM	Over EM	Base F1	Over F1
Short (≤2)	81.4%	79.1%	86.8%	85.5%
Medium (3–5)	77.0%	75.2%	86.7%	85.9%
Long (≥6)	60.4%	60.7%	78.4%	80.3%

reducing accuracy on the short and medium answers. The overall EM is reduced by 1.8% and F1 by 0.8% to achieve better results on long answer examples.

- Short: EM decreases 2.3 points, F1 decreases 1.3 points
- Medium: EM decreases 1.8 points, F1 decreases 0.8 points
- Long: EM increases 0.3 points, F1 increases 1.9 points

One important observation to note is the F1 score improves much more significantly than the EM score. This likely means that the model becomes better at partial matches on these longer answer spans without necessarily improvement in exact matches.

These results show that oversampling is an effective technique when it comes to improving specific subsets of our data. The tradeoff between this improvement against overall model performance is a common experience when mitigating biases present in machine learning models.

6.2 Examples of Improved Predictions

Specific examples are outlined to demonstrate that the oversampling approach improved the predictions made by the model. Table 5 shows three examples of long answers where the model with oversampling achieved an exact match while the initial model was not able to.

In the Tesla example, the base model adds extra words to start of the the identified answer span, while the model with oversampling performs exactly as expected. In the Chris Keates example, the base model does the opposite and adds additional words to the end of the answer span unlike the model with oversampling. The Tesla hours example showcases how the oversampling model not only improve the boundary errors observed before, but can help identify the correct portion of the passage as well.

These examples along with a review of others implies that the base model's bias to shorter answers is primarily observed in missing boundaries of the answer spans and predicting an incorrect shorter span from the passage.

7 Discussion

7.1 Implications of Results

As shown in the results of this project, it is possible to mitigate biases present in models trained on benchmark datasets through targeted improvements on subsets of the data. It is demonstrated that oversampling in particular resulted in a direct increase in performance of longer answer lengths when training with SQuAD. The decision to leverage the default models training architecture or the oversampling approach depends primarily on the intended behavior of the model.

Typically consistent performance across multiple answer lengths would be preferred and in this situation the default benchmark models would seem to be better suited for overall performance. However, if longer answers are common for the specific use case like in domains where detailed explanations are provided, then oversampling may be an effective way to tailor the models to perform better.

Long answer lengths make up a difficult subset of data for models to perform consistently well on. Although, they only make up 9.8% of the dataset used for evaluation, this can be important for critical situations where models need to be effective on these edge scenarios.

7.2 Improvements and Future Directions

As a single-person project, the scope was limited to fit the effort of the analysis and model improvements. However, there are multiple areas for improvement that could be worth investigating further.

- **Hyperparameter Tuning**: The oversampling threshold of 6 and oversampling factor of 3 were both chosen based on the long answer length, but multiple runs with different parameters could have produced different results.
- **Scaling Gradients**: Rather than oversampling, the model could be trained to scale gradients based on the observed answer lengths through approaches like per-example loss weighting. This may also result in less training time for the models as well.
- **Selection Priority**: In the approach taken in this project, all long answer examples are treated the same. Looking at methods that prioritize specific types of these examples which are more challenging to learn could result in better model training without sacrificing overall performance.

8 Conclusion

This project focused on fine-tuning an ELECTRA-small model on SNLI and SQuAD to analyze the performance of models on benchmark datasets with the default training setup. These produced an accuracy of 89.9% for the NLI task along with an EM of 78.1% and F1 of 85.9% for the QA task. Notably, the QA model demonstrated a large difference in the performance of long answer lengths with EM reduced to 60.4% and F1 to 78.4% on this subset.

To mitigate the answer length bias, oversampling was introduced into the training pipeline to increase the number of long answer examples in the training dataset. This was shown to improve EM by 0.3% to 60.7% and F1 by 1.9% to

Table 5. Fixed Long-Answer Predictions with Oversampling

Question	Answer	Base Model	Oversampling Model
What did Tesla do with his patents causing him to lose them?	assigned them to the company in lieu of stock.	he had assigned them to the company in lieu of stock	assigned them to the company in lieu of stock
Before dinner what were Tesla's working hours?	9:00 a.m. until 6:00 p.m. or later	8:10 p.m.	9:00 a.m. until 6:00 p.m. or later
A statement made by Chris Keates caused issues with whom?	child protection and parental rights groups	child protection and parental rights groups. Fears of being labelled...	child protection and parental rights groups

80.3% compared to the default fine-tuning approach. However, this resulted in an overall decrease of 1.8% on EM and 0.8% in F1 for the QA model.

The produced results from this project demonstrate that there can be a tradeoff between overall model performance and targeting improvement on a challenging subset of the data. This is critical to model development as it showcases that overall performance metrics do not entirely capture the effectiveness of a model. Certain use cases may be better served by tailoring the model for these subsets of data at the cost of overall behavior.

There are multiple approaches to reducing bias in machine learning models, but oversampling is shown to be an effective one for certain development scenarios. This project also demonstrates the importance of looking more closely at benchmark model performance through a more comprehensive analysis approach to better identify areas of improvement.

Cite This Work

Subhan Chaudry. 2025. *Analyzing Fine-Tuned ELECTRA Performance on Benchmark Datasets & Mitigating Answer Length Bias.*. Technical Report, University of Texas at Austin.

References

[1] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. (2018). Hypothesis only baselines in natural language inference. *SEM*, 180-191.

[2] Divyansh Kaushik and Zachary C. Lipton. (2018). How much reading does reading comprehension require? A critical investigation of popular benchmarks. *EMNLP*, 5010-5015.

[3] He He, Sheng Zha, and Haohan Wang. (2019). Unlearn dataset bias in natural language inference by fitting the residual. *DeepLo Workshop*, 132-142.

[4] Nelson F. Liu, Roy Schwartz, and Noah A. Smith. (2019). Inoculation by fine-tuning: A method for analyzing challenge datasets. *NAACL*, 2171-2179.

[5] Dheeru Dua, Pradeep Dasigi, Sameer Singh, and Matt Gardner. (2021). Learning with instance bundles for reading comprehension. *arXiv*, arXiv:2104.08293.