Named Entity Recognition as a Lens for Understanding ELECTRA-Small's Performance on QA Tasks

Billy Heidel
University of Texas at Austin, Department of Statistics and Data Sciences
williamheidel@utexas.edu

Abstract

This study explores the performance and limitations of the pre-trained ELECTRA-small model on question answering (QA) tasks using the Stanford Question Answering Dataset (SQuAD) as a benchmark. By analyzing cases where the model produces entirely incorrect answers, we identify a significant reliance on shallow linguistic cues and named entity recognition (NER). To better understand the model's decision-making process, we employ various ablation techniques and train on adversarial datasets, including SQuAD Adversarial, Adversarial QA, and HotpotQA. While these datasets improve performance on challenging examples, overall reasoning capabilities remain constrained. The findings suggest that further improvements in architecture and training objectives, such as incorporating knowledge distillation or alternative loss functions, are necessary for significant advancements in comprehension and entity resolution. This work provides insights into both the strengths and limitations of ELECTRA-small and outlines directions for future research in QA model development.

1 Introduction

1.1 Named Entity Recognition (NER)

Named Entity Recognition (NER) is an NLP subtask that focuses on identifying and categorizing specific entities mentioned in text. These entities typically include names of people, organizations, locations, dates, and other categories relevant to the context. In essence, a named entity represents a real-world object or concept that belongs to one of these predefined categories. For example, the name "Peyton Manning" would fall under the Named Entity category "PERSON," while the number "18" would be categorized as "CARDINAL."

NER has been thoroughly studied for decades. Great strides in this field have been made on documents with well-structured grammar and a limited vocabulary, and current efforts are being made to increase performance on more dynamic and less well-structured texts such as tweets from Twitter (Partalas et al., 2016).

1.2 Error Analysis

This project attempts to analyze the performance of the pre-trained NLP model ELECTRA-small on Question Answering (QA) tasks through the lens of NER.

We elected to use the Stanford Question Answering Dataset, or "SQuAD", for our analysis. SQuAD was created to provide a high-quality benchmark for evaluating and advancing machine reading comprehension systems. It was developed by researchers at Stanford University to address the growing need for standardized datasets that test a machine's ability to understand and extract relevant information from text (Rajpurkar et al., 2016)

Our "Base" ELECTRA-small model was fine-tuned on the SQuAD training dataset (87,599 examples), and we analyzed its performance on the SQuAD validation set (10,570 examples) as a baseline. Its performance was an exact match (EM) of 78.16 and F1 Score of 86.23.

For this analysis, we decided to examine exclusively the errors that the model got completely incorrect (an F1 score of 0): a total of 870 examples out of 2,308 that were at least partially incorrect.

1.3 Incorrect Examples

Upon inspection of these 870 examples, we found that many of the model's predicted answers were very plausible despite being incorrect. Take the following example:

Question: Which player had the most interceptions for the season?

Correct Answer: Kurt Coleman

Predicted Answer: Josh Norman

Not only is the predicted answer a person, but within the context Josh Norman is also a cornerback who had four interceptions that year, compared with Kurt Coleman's seven.

Below is another example where the correct answer is a number:

Question: How many games did the Broncos lose during their regular 2015 season?

Correct Answer: 4

Predicted Answer: 39

This was the case for almost all of the 870 examples inspected. To drill down into these errors even further, we used the Python library spaCy (Honnibal et al., 2020) to quantify how many of these errors could be classified as a case of the model choosing the wrong named entity. The spaCy libary has 18 different named entity groups, which are listed as:

1. CARDINAL: Numerals that do not fall under another type
2. DATE: Absolute or relative dates or periods
3. EVENT: Named hurricanes, battles, wars, sports events, etc.
4. FAC: Buildings, airports, highways, bridges, etc.
5. GPE: Countries, cities, states
6. LANGUAGE: Any named language
7. LAW: Named documents made into laws.
8. LOC: Non-GPE locations, mountain ranges, bodies of water
9. MONEY: Monetary values, including unit
10. NORP: Nationalities or religious or political groups
11. ORDINAL: "first", "second", etc.
12. ORG: Companies, agencies, institutions, etc.
13. PERCENT: Percentage, including "
14. PERSON: People, including fictional
15. PRODUCT: Objects, vehicles, foods, etc. (not services)
16. QUANTITY: Measurements, as of weight or distance
17. TIME: Times smaller than a day
18. WORK_OF_ART: Titles of books, songs, etc.

Out of 870 completely incorrect examples, 452 were identified by spaCy as cases where the model selected the wrong named entity as the answer. This suggests that the model understands the "category" of answer to look for, which begs a critical question: how does the model choose between multiple answers that are reasonable?

1.4 Analysis Using Model Ablations

In an effort to determine which parts of the model lead to decision-making on QA tasks, we ablated different parts of the model and evaluated its performance.

The reasoning behind using model ablations is that by changing or removing parts of the model we can determine that piece's impact on model performance. For example, (Kaushik and Lipton, 2018) fed models corrupted datasets that included only either the passage or question (but not both) and noticed that on most datasets the performance was unexpectedly high, indicating that most information in these datasets were completely unnecessary to obtain the correct answers. In (Chen and Durrett, 2019), the authors fed sentences from the WikiHop and HotpotQA datasets into their model one at a time and used the answer that the model was most confident in to test how many examples really required multi-hop reasoning over multiple sentences to find the correct answer.

1.4.1 Ablations Definitions

We used ablations to test the hypothesis that the model uses clues from the question to determine the type of entity to search for, then scans the context and chooses the named entity whose context is closest to the question. The ablations used in this analysis were:

1. Mask Question Keywords

   Purpose: Test if the model relies on superficial cues from the question (e.g., 'who' or 'when') rather than deeper reasoning.

   Experiment: Replace question keywords like 'who,' 'when,' or 'where' with neutral tokens (e.g., '[KEYWORD]').

   Expected Result: A significant drop in accuracy would indicate over-reliance on shallow keyword-based patterns rather than deeper entity matching.

2. Answer-Only

   Purpose: Determine if the model's predictions are overly influenced by the presence of certain answer types or patterns.

   Experiment: Provide the model with only the answer text (in place of the question) and check whether it predicts the answer correctly.

   Expected Result: If the model frequently predicts the answer correctly, it suggests it might be picking up biases or shortcuts in the dataset (e.g., specific answer spans are more likely for certain questions).

3. Limit Context Length

   Purpose: Determine if the model relies on context proximity for prediction and struggles with longer contexts.

   Experiment: Truncate the context to only the 25 characters before and after the correct answer.

   Expected Result: If accuracy increases sharply, it suggests the model relies on specific parts of the context for prediction.

4. Randomize Context Sentence Order

   Purpose: Test if the model relies more on context structure rather than semantic alignment.

   Experiment: Randomize the order of sentences in the context.

Expected Result: A drop in performance indicates that the model heavily relies on context sequence rather than content relevance.

5. Context Token Shuffling

   Purpose: Check if the model relies on understanding the syntactic structure of the passage.

   Experiment: Shuffle the words within the context while maintaining the original vocabulary.

   Expected Result: If the model still performs well, it may indicate that the model is not truly relying on syntactic and grammatical cues, which are often essential for true comprehension.

6. Nonsense Context

   Purpose: Test if the model relies more on context structure rather than semantic alignment.

   Experiment: Create a context of nonsensical words and randomly add the correct answer into it.

   Expected Result: A relatively high performance indicates that the model heavily relies on entity recognition rather than content relevance.

7. Mask Entity Names in Context

   Purpose: Evaluate if the model is overly reliant on entity names without proper contextual understanding.

   Experiment: Mask all entities in the context (e.g., replace names like 'Bill Gates' with '[MASK]') except the correct answer.

   Expected Result: If the model still performs well, it may indicate that the model is not truly relying on syntactic and grammatical cues, which are often essential for true comprehension, and is instead only searching for an answer of a reasonable entity type.

8. Substitute Entity Names in Context

   Purpose: Evaluate if the model is overly reliant on entity names without proper contextual understanding.

   Experiment: Replace all entities in the context with the correct answer.

   Expected Result: If the model still performs well, it may indicate that the model is not truly relying on syntactic and grammatical cues, which are often essential for true comprehension, and is instead only searching for an answer of a reasonable entity type.

9. Add Answers and Question to Context

   Purpose: Evaluate if the model is overly reliant on similarities in words and structure between the context and the question in making its predictions.

   Experiment: Add the answer after the question and randomly put it somewhere in the context.

   Expected Result: If model performance improves, the model is likely over-emphasizing vocabulary and grammatical similarities between the context and the question.

1.4.2 Ablations Results

After performing each ablation on the SQuAD validation dataset then evaluating the base model's performance on this "corrupted" dataset, performance was as follows:

Ablation	EM	F1
Baseline (no Ablation)	78.16	86.23
Add Answers and Question to Context	91.01	94.61
Limit Context Length	77.64	86.53
Substitute Entity Names in Context	77.42	86.36
Randomize Context Sentence Order	76.29	84.60
Nonsense Context	75.01	84.90
Mask Entity Names in Context	74.50	82.56
Mask Question Keywords	62.58	72.76
Context Token Shuffling	9.23	18.93
Answer-Only	7.71	22.51

Table 1: Results of Base Model after Ablations (Ordered by F1).

The biggest standout from these results was that the "Add Answer and Questions to Context" ablation improved the performance to almost perfect with a 90.01 EM and 94.61 F1 score. This supports the hypothesis that the model is scanning the passage for word and syntax similarity to the question, then chooses a reasonable answer from that span of the expected entity type.

Another notable result is that the "Nonsense Context" ablation performed almost as well as the baseline, which is staggering because the corrupted context contained completely nonsensical words except for the randomly inserted correct answer. This suggests that the model strongly emphasizes words in its own learned vocabulary regardless of context.

The ablations related to entity names ("Mask Entity Names in Context" and "Substitute Entity Names in Context") also performed very similarly to the base dataset. This behavior is challenging to explain; perhaps the most reasonable explanation is that the NER model from spaCy, which was used to recognize and replace the named entities in each example, is not fully equipped to recognize the named entities in these passages. Hopefully future improvements to the model will lead to a more complete recognition of named entities such as those in the SQuAD dataset.

Performance did have a significant drop when the "Mask Question Keywords" ablation was applied. We can infer from this result that the model takes queues from these keywords ('who', 'when', 'what', 'where', etc.) as to what "type" (or named entity category) of answer to return, but it is not

overly dependent on these queues as both the EM and F1 were still over 60.

Similar performance between baseline and the "Limit Context Length" and "Randomize Context Sentence Order" ablations indicates that the model is effective at finding the relevant span in a longer context. When the context is completely jumbled at the token level, like in the "Context Token Shuffling" ablation performance was extremely low. These are all signs that the model is properly able to scan the context and find the correct span where the result is located.

The model's behavior with the "Answer-Only" was quite insightful as well. As a general rule, the model chose a larger span (about sentence-level) that included the correct answer. In one example, when 'Denver Broncos' was fed into the model as the question, the predicted answer was: "Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers". This suggests that the model makes its decision by finding a span of the context that is most similar to the question, then uses clues from the question to determine what type of answer to return. In the example above, the model sees "Denver Broncos" and focuses in on that span, but because there are no other clues from the question (such as 'who') the model does not know what type of answer to return and ends up returning the entire span.

1.5 Summary of Analysis

Through visual inspection of errors and evaluation of model ablations, we posit that the model makes its decision by narrowing down the context based on its similarity to the question and then using clues from the question to pick the correct "type" of answer from this narrowed span. This strategy seems to generally be very effective on the SQuAD dataset, but it can lead to mistakes when the correct context span is less similar to the question than another span in the same context.

2 Approach

2.1 Data Sources

We investigated the model's performance on a handful of challenging/adversarial datasets to assist the model in obtaining a more sophisticated decision-making process.

2.1.1 SQuAD Adversarial

SQuAD Adversarial was introduced by (Jia and Liang, 2017) as a means to confuse models by introducing noise to the text. The "noise" here is not random nonsense, but rather sentences that seem to be plausible but can distract a model from the truth. A few algorithms were tested to create this dataset, and the one we used on our model was the "AddSent" adversary, which mutates the original question, creates a fake answer, and combines them into a sentence that is added to the context:

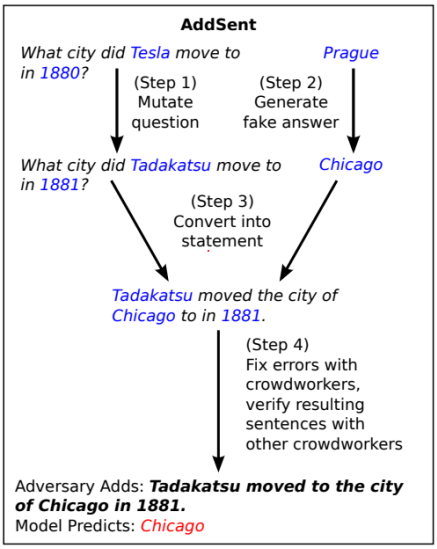

Figure 1: Example of how a question in Adversarial SQuAD is created (Jia and Liang, 2017).

This is exactly the type of adversary we want our model to be able to answer correctly, so the 3,560 examples were added to our dataset during training. Note that this is significantly smaller than the 87,599 training examples in the original SQuAD dataset, so we upsampled the SQuAD Adversarial dataset during training.

2.1.2 Adversarial QA

Adversarial QA (Bartolo et al., 2020) was a dataset designed to fool state-of-the-art QA models at the time it was created, and is based off of the SQuAD 1.1 dataset. What makes this dataset especially interesting is that it was hand-crafted by people rather than being corrupted or generated by A.I. This is particularly impressive considering that it contains 30,000 training examples, 3,000 validation examples, and 3,000 test samples.

This dataset is split into three equal-sized groups, each targeting a particular model: BiDAF, BERT-Large, and RoBERTa-Large. We examined

the performance of each of these subsets separately as well as combined during training to find the best possible combination for improving performance on our most difficult questions.

2.1.3 HotpotQA

The dataset HotpotQA (Yang et al., 2018) is a QA dataset designed explicitly for multi-hop reasoning, which requires a model to use logic to combine multiple pieces of information to come to the correct answer. Here is one example from the HotpotQA Dataset:

> *Question:* Which magazine was started first Arthur's Magazine or First for Women?
>
> *Sentence 1:* "Arthur's Magazine (1844–1846) was an American literary periodical published in Philadelphia in the 19th century."
>
> *Sentence 2:* "First for Women is a woman's magazine published by Bauer Media Group in the USA. The magazine was started in 1989."
>
> *Correct Answer:* Arthur's Magazine

Note that the question is not directly answered by these sentences. For a model to get the correct answer, it must do more advanced reasoning than a simple similarity comparison between the context and question. We trained our model on this dataset in the hope that this sophisticated reasoning can be learned and used to solve many of the mistakes from our base model.

HotpotQA contains two datasets, "Distractor" and "FullWiki". "Distractor" provides a context of 2 relevant paragraphs required to answer the question and 8 "distractor" paragraphs that are unrelated. "Fullwiki" contains the full Wikipedia articles that were used to create the question/context pair, and is therefore more challenging because it requires the model to parse through a much larger context to find the correct answer. Our model was trained and tested on both datasets, over 90k examples each.

2.2 Individual Dataset Performance

We fine-tuned ELECTRA-small on each subset of each of these three datasets and evaluated their performance on the SQuAD validation dataset to see which datasets would be the best for improving our model. The results were as follows:

Dataset/Subset	Full	Wrong	W.E.
SQuAD (Baseline)	78.2/86.2	0/0	0/0
Adversarial SQuAD	6.1/14.0	5.3/13.0	3.7/11.2
Adversarial QA	36.6/51.7	13.7/24.4	11.7/22.5
AQA (BiDAF)	10.1/21.0	6.2/15.8	5.1/14.5
AQA (BERT)	6.4/16.6	3.8/12.0	3.6/12.2
AQA (RoBERTa)	5.5/17.3	2.7/11.5	2.3/11.7
HPQA (Distractor)	59.8/69.7	17.9/25.8	16.0/23.1
HPQA (FullWiki)	59.9/69.8	18.6/26.1	16.8/23.8

Table 2: Results of training the model on each dataset. Evaluated on SQuAD eval.

Here "Full" refers to the full SQuAD validation dataset of 10,570 examples, "Wrong" refers to the 870 examples from the SQuAD validation dataset whose prediction from our Base model had an F1 of 0, and "W.E." refers to the 452 of these "Wrong" examples whose predicted answer was a "Wrong Entity" of the same type.

These results show that Adversarial SQuAD does a particularly poor job of solving the true SQuAD dataset; this is likely due to the extremely small size of the Adversarial SQuAD. The full Adversarial QA dataset performed significantly better than any of its three subsets; this is likely again a function of dataset size. Both HotpotQA datasets performed the best, with the "FullWiki" dataset just barely outperforming the "Distractor" dataset.

2.3 Training Combinations with the SQuAD Dataset

We tested various combinations of Adversarial SQuAD ("ASQ"), Adversarial QA ("AQA"), and HotpotQA ("HPQA") with the original SQuAD dataset to see which performed the best on our difficult examples and the dataset as a whole. Because of the varying sizes of these datasets, we also tested combinations with different proportions of each. We always used 100% of the SQuAD training dataset (proportion '1.0'), and we added proportions from 0% (proportion '0') to 300% (proportion '3.0') for some of the relatively smaller datasets like Adversarial QA. Because the HotpotQA dataset is about the same size as SQuAD, we tested combinations from 10% to 100%.

SQuAD	ASQ	AQA	HPQA	Full
1.0	0	0	0	78.2/86.2
1.0	1.0	0	1.0	77.2/85.0
1.0	2.0	0	0	76.4/84.5
1.0	1.0	0	0	75.9/84.2
1.0	0	1.0	0.50	75.8/84.1
1.0	0	0	1.0	75.4/84.0
1.0	0	1.0	0.25	75.6/83.9
1.0	0	1.0	0.1	75.4/83.8
1.0	0	0	0.5	75.6/83.7
1.0	0	1.0	0	75.3/83.7
1.0	0	0	0.25	75.3/83.6
1.0	0	2.0	0	74.7/83.3
1.0	0	0	0.1	74.4/83.1
1.0	0	3.0	0	73.7/82.5

Table 3: Results of training the model on a combined proportion of each dataset. Evaluated on SQuAD eval.

SQuAD	ASQ	AQA	HPQA	Wrong
1.0	1.0	0	1.0	23.6/28.7
1.0	2.0	0	0	22.6/28.4
1.0	0	0	1.0	20.4/26.6
1.0	0	2.0	0	20.4/26.0
1.0	0	1.0	0.50	19.9/26.0
1.0	0	3.0	0	18.4/24.9
1.0	0	0	0.1	18.1/24.5
1.0	0	1.0	0.25	19.0/24.4
1.0	1.0	0	0	18.3/24.4
1.0	0	0	0.25	18.4/24.0
1.0	0	1.0	0.1	18.6/23.9
1.0	0	1.0	0	18.8/23.7
1.0	0	0	0.5	17.9/23.7

Table 4: Results of training the model on a combined proportion of each dataset. Evaluated on SQuAD eval examples that the Base Model got 100% wrong.

SQuAD	ASQ	AQA	HPQA	W.E.
1.0	2.0	0	0	24.3/29.4
1.0	1.0	1.0	0	24.6/29.0
1.0	0	0	1.0	20.0/26.3
1.0	1.0	0	0	220.6/26.2
1.0	0	1.0	0.50	18.3/24.6
1.0	0	3.0	0	17.7/24.1
1.0	0	2.0	0	18.0/24.0
1.0	0	0	0.25	17.5/23.5
1.0	0	0	0.5	16.3/22.6
1.0	0	1.0	0.25	16.3/22.4
1.0	0	1.0	0	16.8/22.3
1.0	0	0	0.1	15.4/22.0
1.0	0	1.0	0.1	15.3/21.3

Table 5: Results of training the model on a combined proportion of each dataset. Evaluated on SQuAD eval examples where the Base Model chose the wrong entity.

Unfortunately, these datasets did not lead to better performance on the overall SQuAD validation dataset. This is not particularly surprising, however, as including different datasets inherently forces the model to "pay less attention" to the original SQuAD dataset. However, training on these Adversarial datasets did increase performance on the most difficult examples, with our "Best" model (SQuAD, 100% of Adversarial SQuAD, and 100% of HotpotQA), getting approximately 25% of these examples correct. This "Best" model also performed slightly better on examples where the original model selected the wrong entity than on all incorrect examples, with an EM/F1 of 24.6/29.0 compared to 23.6/28.7. This is slightly surprising because none of the Adversarial datasets individually performed better on "W.E" examples than "Wrong" examples; perhaps the model learned something within the combination of datasets that it couldn't learn individually.

3 Results

3.1 Ablations on the Updated Model

We performed the same ablations from our original analysis on the new "Best" model to see if there were any meaningful changes in behavior:

Ablation	EM	F1
Baseline (no Ablation)	77.20	85.00
Add Answers and Question to Context	89.79	93.88
Nonsense Context	79.82	88.80
Limit Context Length	77.24	86.88
Substitute Entity Names in Context	75.64	84.54
Randomize Context Sentence Order	75.18	83.51
Mask Entity Names in Context	73.91	81.64
Mask Question Keywords	59.43	69.75
Context Token Shuffling	9.71	19.52
Answer-Only	6.52	17.69

Table 6: Results of the "Best" Model after Ablations.

Behavior here is very similar to the Base model, but there is one exception: the "Nonsense Context" ablation actually performed better than the base dataset. This suggests that the "Best" model places even more emphasis on its recognition and understanding of individual words than the original context. Even though this was not the change in reasoning we were hoping to achieve, it ended up being fairly effective at answering the most challenging examples, especially those related to wrong entities.

3.2 Changes from the Original Dataset

Our "Best" model made slightly more mistakes than the "Base" model overall:

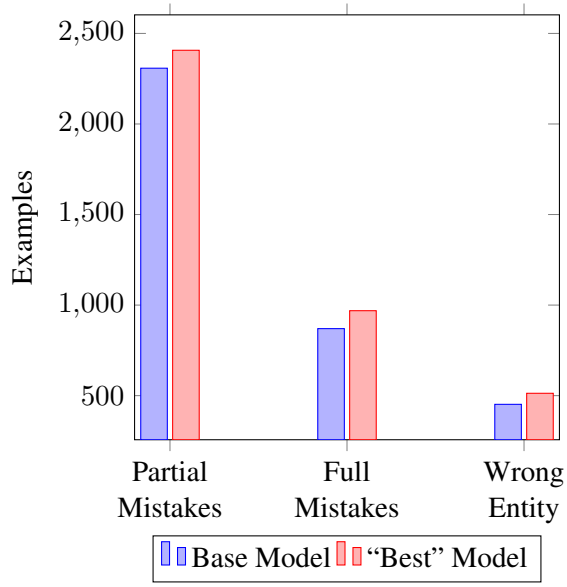

Figure 2: A comparison of mistake types between the "Base" and "Best" models. The categories represent partial mistakes, full mistakes, and wrong entity selections.

We define "Partial Mistakes" as mistakes with an F1 less than 1, "Full Mistakes" as mistakes with an F1 $= 0$, and "Wrong Entity" as mistakes where the model chose the wrong named entity.

To took deeper into how the "Best" model's mistakes compared with that of the "Base" model, below are a few examples of questions from the SQuAD validation set and each model's predicted answer.

First, we have a question that both models got correct:

> *Question:* What is the AFC short for?
>
> *Correct Answer:* American Football Conference
>
> *"Base" Model Answer:* American Football Conference
>
> *"Best" Model Answer:* American Football Conference

Next, a question that the "Best" model got correct that the "Base" model did not:

> *Question:* What was the theme of Super Bowl 50?
>
> *Correct Answer:* golden anniversary
>
> *"Base" Model Answer:* an American football game
>
> *"Best" Model Answer:* golden anniversary

Then, a question that the "Base" model got correct that the "Best" model did not:

> *Question:* Which NFL team represented the NFC at Super Bowl 50?
>
> *Correct Answer:* Carolina Panthers
>
> *"Base" Model Answer:* Carolina Panthers
>
> *"Best" Model Answer:* Denver Broncos

and finally, a question that neither model got correct:

> *Question:* What is the seldom used force unit equal to one thousand newtons?
>
> *Correct Answer:* sthène
>
> *"Base" Model Answer:* the metric slug
>
> *"Best" Model Answer:* mass

Performance between these two models was overall very similar. Below is the transition matrix for errors between the "Base" model and the "Best" model:

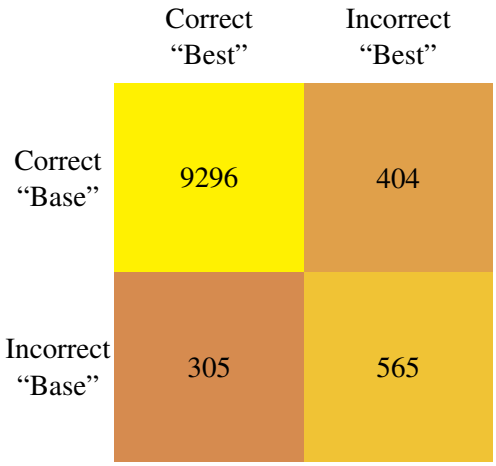

Figure 3: Heatmap showing transitions between the “Base” and “Best” models. Each cell value indicates the count of validation examples in each transition.

4 Conclusion

Out of the three adversarial datasets examined, HotpotQA was the best on model performance overall and on challenging examples where the model has to choose between multiple named entities. The addition of the Adversarial SQuAD dataset to a combination of SQuAD and HotpotQA showed a small but significant performance on the SQuAD validation dataset, indicating that the modest-sized dataset has value for “filling in gaps” in language comprehension that the two other datasets do not have. The inclusion of the Adversarial QA dataset did not show any noteworthy positive impact on model performance.

Incorporating additional datasets can improve ELECTRA-small’s performance on specific questions in the SQuAD dataset, but fails to improve model reasoning in a significant way. Changes in performance from the “Base” model to the “Best” model seem to stem from a change in emphasis of certain key words in the context or question rather than from a fundamental change in language comprehension. This suggests that changing model architecture might be more effective in improving how the model reasons about choosing between two named entities from a passage. A future project involving changes to the number and/or size of the model’s transformer layers, incorporating knowledge distillation from another model, and/or changing the training objective (e.g. from cross-entropy loss to focal loss) could potentially improve the model at a more fundamental level, leading to a more substantial increase in performance than what this project was able to accomplish.

Acknowledgments

A huge thank you to Professor Durrett, the TA’s, and anyone who took the time to read about my work. Thank you for helping me learn NLP!

References

Max Bartolo, Alastair Roberts, Johannes Welbl, Sebastian Riedel, and Pontus Stenetorp. 2020. Beat the AI: Investigating adversarial human annotation for reading comprehension. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8521–8535, Online. Association for Computational Linguistics.

Jifan Chen and Greg Durrett. 2019. Understanding dataset design choices for multi-hop reasoning. 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 4026–4032, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python. Version 2.3.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics.

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

Ioannis Partalas, Cédric Lopez, Nadia Derbas, and Ruslan Kalitvianski. 2016. Learning to search for recognizing named entities in twitter. In *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*, pages 171—177. The COLING 2016 Organizing Committee.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380. Association for Computational Linguistics.