NLP Final Project

Anonymous Individual Project

Abstract

In this paper, I evaluate the performance of an ELECTRA-small model on the Stanford Natural Language Inference (SNLI) dataset. The performance analysis focuses on identifying dataset artifacts by using two published methodologies. The first method is the "consistency framework" presented by Gardner et al. (2021), and the second method is "contrast sets" in Gardner et al. (2020). Next, I attempt to improve the model performance by identifying "forgettable examples" as described in Yaghoobzadeh et al. (2021), and fine-tuning the ELECTRA model on the forgettable examples. I show that the fine-tuning results in better performance on the forgettable examples, but not on the entire dataset.

1 Introduction

The current state-of-the-art performance on the SNLI dataset is 93% accuracy (https://paperswithcode.com/dataset/snli). However, while the accuracy is high, this does not always mean that the model has learned about the language concepts in the dataset. Specifically, the model could be picking up on dataset artifacts, that is spurious correlations in the data. When the model is not evaluated on data without these artifacts, it is likely that the model will not perform as well. I investigate the performance of an ELECTRA-small model on the SNLI dataset, identify artifacts, and attempt to mitigate the impacts of the artifacts.

2 Dataset

The Stanford Natural Language Inference dataset (Bowman et al., 2015) consists of pairs of sentences, where the first sentence is the premise, and the second sentence is the hypothesis. The relationship between the premise and hypothesis is manually labeled by annotators as entailment, contradiction, or neutral. The label with the majority vote of the annotators is the gold label in the dataset. If the annotators did not reach a consensus, the gold label is "-". When the SNLI dataset is obtained through the project starter code, the "-" gold labels have already been filtered out from the dataset. The source of the premise sentences are captions taken from Flickr images. The hypothesis sentences were generated from annotators who were tasked with generating a hypothesis sentence that entailed, contradicted, or was neutral for the premise sentence. The dataset contains 550,152 training examples, and 10,000 examples each in the test and validation splits. Example sentences are shown in Table 1 taken from https://paperswithcode.com/dataset/snli.

Previous work has found artifacts in the SNLI dataset. Gururangan et al. (2018) and Poliak et al. (2018) show that having "cat", "sleeping", or "not" in the premise or hypothesis occurs more in contradiction examples. Additionally, McCoy et al. (2019) and Zhang et al. (2019) showed that having words in both the hypothesis and premise is highly correlated with entailment.

2 Model Training

I trained an ELECTRA-small model on the SNLI dataset using an NVIDIA GeForce GTX 1050 Ti GPU. This dataset has 3 classes, where

the 0 label denotes entailment, 1 denotes neutral, and 2 denotes contradiction. I used the default

Text	Judgments	Hypothesis
A man inspects the uniform of a figure in some East Asian country.	contradiction C C C C C	The man is sleeping
An older and younger man smiling.	neutral N N E N N	Two men are smiling and laughing at the cats playing on the floor.
A black race car starts up in front of a crowd of people.	contradiction C C C C C	A man is driving down a lonely road.
A soccer game with multiple males playing.	entailment E E E E E	Some men are playing a sport.
A smiling costumed woman is holding an umbrella.	neutral N N E C N	A happy woman in a fairy costume holds an umbrella.

Table 1: Example sentences from the SNLI data (Source: https://paperswithcode.com/dataset/snli). The column labeled "Text" is the premise sentence, and the column labeled "Judgments" shows the five individual annotators' decisions on the relation of the hypothesis to the premise, and above the five choices is the consensus choice.

parameters for training: `max_length`=128, `num_train_epochs`=3, and `per_device_train_batch_size`=8. The run time for the training is 9:09:17 and the final training loss is 0.415. On the evaluation dataset, the model achieves 89% accuracy with a loss of 0.39. The loss and accuracy of the model on the eval (validation) dataset for each epoch is shown in Figure 1.

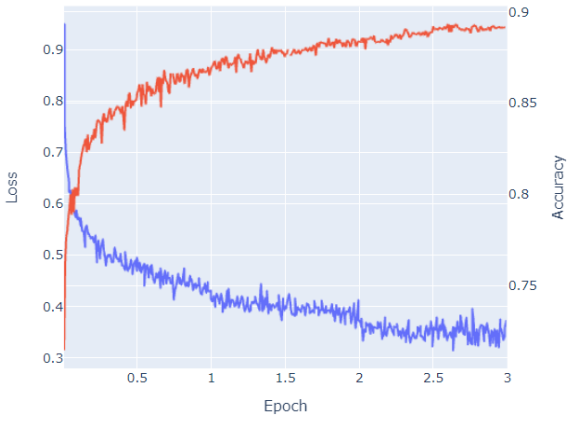

Figure 1: Loss and accuracy vs. epoch for ELECTRA model on eval dataset.

3 Eval Set Performance of ELECTRA

First, I look at the performance of the model's eval predictions using a confusion matrix, shown in Figure 2. The confusion matrix shows that the distribution of classes in the eval set is roughly the same, in other words each class has roughly one-third of the examples. The raw counts for each class are 3329, class 0, 3235, class 1, and 3278, class 2. By examining the confusion matrix, we can see that for the 0 and 2 classes, entailment and contradiction, when the model makes an incorrect prediction, it more often predicts the neutral class, rather than the opposite class.

Specifically, for the entailment class, 17% of the incorrect predictions are for the contradiction class, and for the contradiction class, 26% of the incorrect predictions are for the entailment class. However, for the neutral class, the model is about equally likely to predict entailment or contradiction for the incorrect prediction. The actual percentage is 51% predicted contradiction when the gold label is neutral. Furthermore, the model successfully predicts 91% of the entailment gold labels, 86% of the neutral gold labels, and 91% of the contradiction gold labels. So it appears that the

model has the most difficulty with the neutral class.

4 Consistency Framework Analysis

4.1 Hypothesis Test

The first method I used to analyze the dataset is the statistical test framework in Gardner et al. (2021). I repeated the analysis done in this paper, since I could not find the code for the paper online. The authors of this paper argue that for language understanding tasks, no one feature should give information about the gold label. "Competency problems" are problems where $p(y|x_i)$ is uniform over the class label y for each feature x_i. The example given in the paper is the word "amazing" in movie reviews. The authors claim that someone can use "amazing" in contexts without positive sentiment, so they would not predict a review to have positive sentiment based on amazing on its own. Instead, it is the interaction among features which leads to the sentiment classification. While a review that contains "amazing" is more likely to be positive, that is because speakers of the language more often choose to use "amazing" in a positive review, which is a distributional effect. Additionally, if a model learns a dataset's individual feature correlations, it learns about the human biases in constructing the dataset, and is not truly learning meaning in the words.

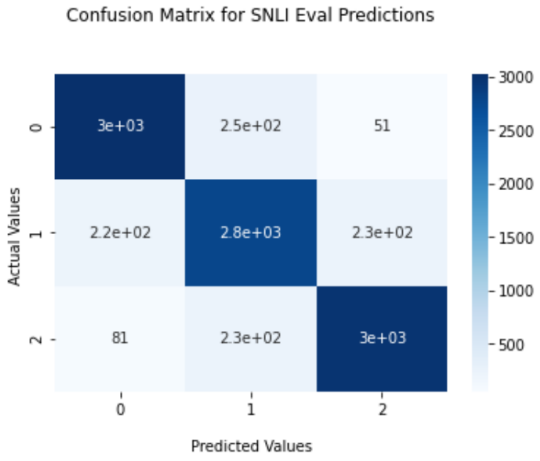

Figure 2: Confusion matrix of ELECTRA model trained for 3 epochs evaluated on the eval set.

The authors derive a hypothesis test to determine if there is enough evidence to reject the null hypothesis that the data is unbiased. First, I repeat the hypothesis test and reproduce their Figure 1. To do this, I read in the "train" split of the SNLI dataset. The paper does not specify which split of the SNLI dataset it uses for the hypothesis test. Additionally, the paper does not specify whether the words in only the hypothesis text, premise text, or in both hypothesis and premise texts are looked at. As described in the paper, I removed punctuation and tokenized the hypothesis and premise on whitespace. I count each time a word appears in either the hypothesis or the premise. Using both the premise and hypothesis sentences results in a plot similar to the paper's Figure 1, that is displayed in Figure 3. (I first used only the premise, and this resulted in the word "animal" having n of about 1000, and $\hat{p}$ of 0.33 for each class, which is different from the probabilities and count of "animal" shown in the plot. By using both the premise and hypothesis, this gives n of about 2000 for animal and $\hat{p}$ of 0.56 for entailment, which looks closer to the plot's values.)

To compute the blue line shown in the plot, I first found the z-critical value, z^*. The z critical value for a one-sided test with p_0=⅓ and α=0.01/28,000 is 4.957. Using the values of z^* and p_0, I rearrange equation 1 in the paper to solve for $\hat{p}$, which gives $\hat{p}$ as a function of n on the horizontal axis. This is the equation that I used to make the blue line in my plot. (Note that I did not plot points below the line in gray, as was done in the paper. Additionally, though there is a color scale for a legend, the only colors in the plot are dark purple for 0/entailment, red for 1/neutral, and yellow for 2/contradiction.)

2% of words have a statistically significant z value (a z value greater than z^*) for entailment, 11% have a statistically significant z value for neutral, and 6.8% have a statistically significant z value for contradiction.

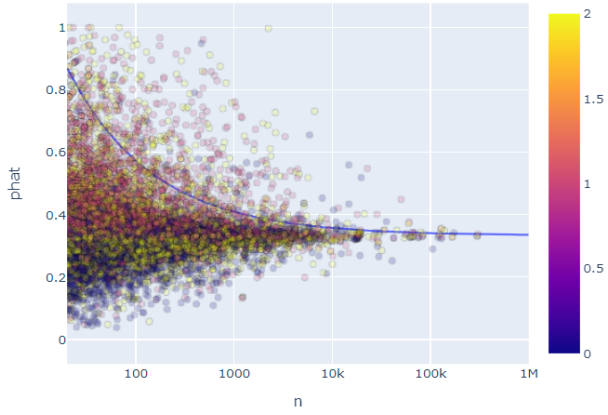

Figure 3: Features (1-grams) which have a statistically significant correlation with the class label are above the blue line. The number of occurrences of each feature is plotted against the conditional probability of the label given the feature.

Figure 4 shows word clouds for each class with the top 50 words for each class having the largest z scores. The size of the word in the word cloud is proportional to the word's z score.

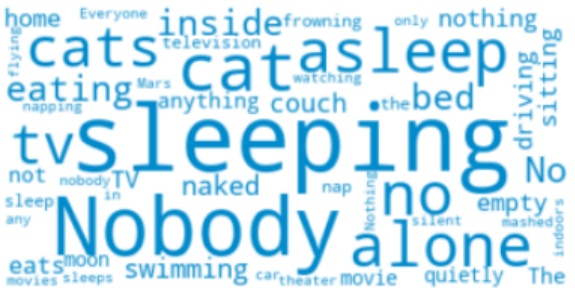

Figure 4: Words with largest z scores for contradiction class.

Figure 4: Words with largest z scores for entailment class.

Figure 4: Words with largest z scores for neutral class.

4.2 Model Analysis

Now that I've found individual word artifacts, I follow the authors' analysis to determine whether the NLP model has learned to bias its predictions based on the artifacts. First, I filter out words (1-grams) which occur less than 20 times in any class. Then I assign each word to the class where its z-score is the highest. In each class, I take the words with the 50 highest and 50 lowest z-scores and make a new dataset. For

each word, the dataset has two examples. The first example has a premise that is equal to the word, and a hypothesis that is an empty string. The second example has a premise that is an empty string, and a hypothesis that is equal to the word. Next, I make predictions on this new dataset with the trained ELECTRA model. The accuracy on this data is 46%. We would expect 33% accuracy, on average, if the single words gave no information about the label and the model was randomly guessing.

For each pair of examples, I average the predicted probabilities to get $p(y|x_i)$. Then I compute

$$\Delta\tilde{p}_y = \sum_{x_i \in \mathcal{X}_{y,z^*_>}} \tilde{p}(y|x_i) - \sum_{x_i \in \mathcal{X}_{y,z^*_<}} \tilde{p}(y|x_i).$$

(the difference between the average model predicted class probability between large z^* tokens and low z^* tokens). The results are displayed in Table 2.

Class	$\Delta\, p_y$
entailment	-0.2%
neutral	+7.1%
contradiction	+11.7%

Table 2: Average model predicted class probability for one word between large z^* tokens and low z^* tokens. These results show that the model has bias for neutral and contradiction. The original paper found significant bias in the entailment class using the ALBERT model.

5 Contrast Sets

A contrast set is a constructed set of inputs that are similar to existing inputs, but usually have a different label. A metric used to evaluate the performance of a model on a contrast set is the "contrast consistency", which measures whether the model correctly predicts the gold label of all the contrast set data points. A contrast set differs from adversarial examples in that the gold label of an adversarial example is the same, but the input changes so that the model makes a different prediction from the initial example. While the authors recommend manual construction of contrast datasets, I have used an existing library that automatically creates contrast sets. I used an SNLI contrast set from the GitHub repo leo-liuzy/LIT_auto-gen-contrast-set. I evaluated the performance of the three epoch ELECTRA model on the test and dev contrast sets. The performance on these contrast sets is 74% and 76% respectively, which is significantly lower than the 89% on the eval set of the original data. This is additional evidence that artifacts are affecting the model performance.

6 Fix: Forgettable Examples

Yaghoobzadeh et al. (2021) uses the concept of example forgetting to find examples which contradict spurious correlations in the training dataset. An example is forgettable if during the training process, it is never correctly classified, or if it is correctly classified but then becomes misclassified later in training. Forgetting occurs because when a stochastic gradient descent step occurs, the update can cause worse performance on some examples. The authors claim that using shallower networks than the final model to find the forgettable examples, and then fine-tuning the model using the forgettable has better performance than fine-tuning on forgettables from the original model. Additionally, they found that training the final model only on forgettable examples, rather than the entire training set results in worse performance.

I used code in Yaghoobzadeh et al. (2021) github repo: `github.com/sordonia/hans-forgetting`. Since the authors did not evaluate the SNLI dataset. I modified the

code to run on SNLI data. Additionally, their code depends on the `apex` python library. Using this library as-is gave me errors while running the repo code. So I additionally modified one of the `apex` python files as shown in the issue https://github.com/NVIDIA/apex/pull/1282. As described in the paper, I ran two shallow models for 5 epochs. Five epochs of the LSTM model took 1:22:51, five epochs of BoW took 59:15. The construction of the BoW classifier encodes the premise and hypothesis separately as the mean of the word embeddings. The LSTM model is a bi-directional siamese LSTM.

The performance of the models on the eval SNLI dataset is shown in the table below.

Epoch	BoW	LSTM
0	0.330	0.344
1	0.828	0.773
2	0.840	0.790
3	0.846	0.798
4	0.844	0.798
5	0.846	0.798

Table 3: Eval set accuracy vs epoch for BoW and LSTM models defined in sordonia/hans-forgetting repo.

With these models, I computed statistics on which examples were forgotten over the course of model training. I didn't run on the BERT model since it is similar to the ELECTRA model, and we want a shallow model. (The BoW model architecture used has 560K and LSTM has 2M parameters. An ELECTRA-small model has 14M parameters. The BERT-base model that is used in the paper has 110M parameters.) The figure below shows the distribution of the forgetting events. The forgettable examples are mostly those which never are learned during training or have only one forgetting event.

Figure 5: Distribution of number of forgetting events for each model used.

The number of forgettable examples is 82,606 for the LSTM model, and 97,479 for the BoW model. There are 47,786 examples in common between both models. For the LSTM model, 45% of the forgettable events are neutral, and 27% are contradiction and also entailment. For the BoW model, 44% are neutral, and 29% are entailment, while 27% are contradiction. This is similar to the initial ELECTRA model where the model had the worst performance on the neutral class.

I now fine-tune the 3 epoch ELECTRA model on the forgettable examples with a smaller learning rate. I used a learning rate of 5e-6 as the authors of the paper used in their fine-tuning. The authors fine-tuned for 3 epochs, but I fine-tuned for 5 epochs to observe the performance across a longer timeframe. I did this for the BoW and LSTM forgettables separately. The fine-tuning results in better performance on the forgettable examples than the initial model had (about 60% accuracy) but the performance on the eval split of the original dataset starts decreasing as soon as the fine-tuning starts. This is the same for both the BoW and LSTM forgettables fine-tuned model. See Figure 5 for the BoW forgettables fine-tuned

performance. Yaghoobzadeh et al. (2021) did not evaluate their method on the SNLI dataset, but they did on the MNLI dataset and found that fine-tuning on forgettable examples did not improve overall performance of their network. If I had more time for the project, I would also look at forgettables of the 3 epoch ELECTRA model and see if fine-tuning on those examples results in better performance. Additionally, I would explore different learning rates in the fine tuning process.

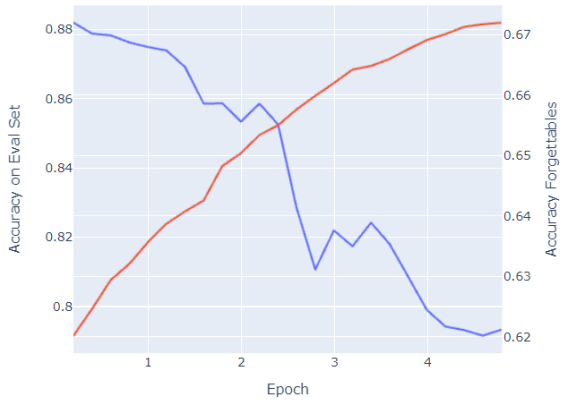

Figure 6: Accuracy on eval set of original SNLI data vs accuracy on forgettables vs epochs for fine-tuned ELECTRA model.

7 Conclusion

In this project, I found dataset artifacts using the consistency framework method, and investigated performance on a contrast set designed to exploit dataset artifacts. To try to improve the performance of the ELECTRA model and mitigate the effects of the artifacts, I used a method proposed in Yaghoobzadeh et al. (2021) to find forgettable examples and fine-tune the model on these examples. While this method did not improve the performance of the model, other ideas might, as discussed in the previous section.

References

[Bowman et al.2015] 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, pages 632–642, Lisbon, Portugal, September. Association for Computational Linguistics.

[Gardner et al.2020] Matt Gardner, Yoav Artzi, Victoria Basmova, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hanna Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating models' local decision boundaries via contrast sets.

[Gardner et al.2021] Matt Gardner, William Merrill, Jesse Dodge, Matthew E Peters, Alexis Ross, Sameer Singh, and Noah Smith. 2021. Competency problems: On finding and removing artifacts in language data. arXiv preprint arXiv:2104.08646

[Gururangan et al.2018] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

[McCoy et al. 2019] 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.

[Poliak et al.2018] Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantic*s, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

[Yaghoobzadeh et al.2021] Yadollah Yaghoobzadeh, Soroush Mehri, Remi Tachet des Combes, T. J. Hazen, and Alessandro Sordoni. 2021. Increasing robustness to spurious correlations using forgettable examples. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3319–3332, Online, April. Association for Computational Linguistics.

[Zhang et al.2019] Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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 1298–1308.