Using Data Artifacts and Dataset Cartography to Improve NLI performance by Training on Hard Subsets

Brian Naklycky

Abstract

Our goal is to find Dataset Artifacts and hard examples using Dataset Cartography and focus our models learning on those examples. We use the SNLI dataset from (Bowman et al., 2015) to test techniques discussed in (Gardner et al., 2021) on finding data artifacts as well as using dataset cartography (Swayamdipta et al., 2020) to improve our performance on hard subsets of the SNLI dataset.

1 Introduction

When training a model you want to ensure that it is learning the task you want it to learn and not correlations between data. Take for example a sentiment analysis model that spuriously learns from the dataset that certain words that correlate with certain sentiments. Here a model could learn that the word 'good' would result in a positive sentiment. With this comes obvious pitfalls in model performance.

Another issue surfaces from training examples that a model struggles with. In a normal training epoch there might be some tough samples that the model particularly struggles with. Learning which examples these are can be beneficial to improving model performance and model interpretability by shedding light on its inner workings and increasing model transparency (Lipton, 2017).

With these issues in mind we use the Electra-small model (Clark et al., 2020) trained on the SNLI dataset. We wrote all the code to train, test, and analyze the model. We found artifacts in the SNLI dataset and used it in conjunction with hard examples that the model struggled with in training. We then trained the model on a hard subset of the training data.

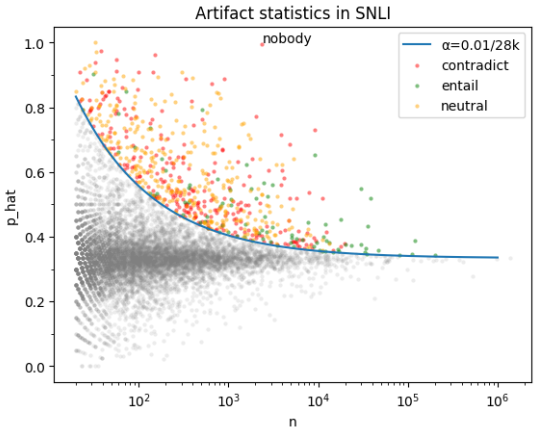

Figure 1: Artifact Statistics in SNLI

2 Methods

For part 1 of this project we were able to fully recreate the Dataset Artifact graph (see figure 1), which shows the number of statistically significantly biased samples in the dataset.

We analyzed the dataset following the statistical test using spurious n-grams/artifact method described in the Gardner et. al. 2021 paper. We collected samples that have words that disproportionately appear in one class in the training data as well as words that have a near equal distribution. In the paper they called these 'artifacts' and tested to see whether or not our model would learn the spurious correlation between the words and the classes in the dataset.

We tested according to procedure described in Gardner et al. 2021, where we took the average of two predicted training samples. The first sample was with the target word as the premise and an empty string as the hypothesis, and then vice versa. This method was used to estimate the underlying distribution of each word, effectively giving us an idea of the model's bias. This was motivated by the fact that the model should not have

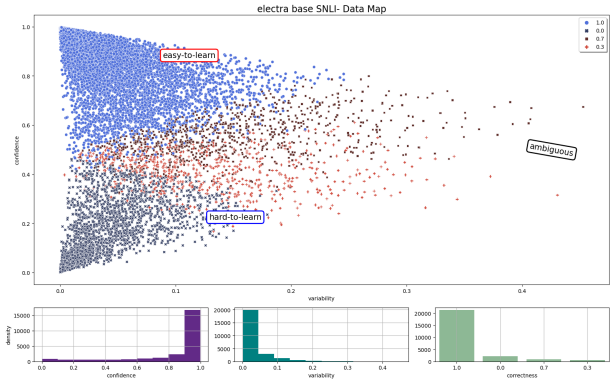

Figure 2: SNLI Dataset Map

any class prediction on a single word. The task should be non-markovian. However, the model might be learning correlation if a word that appears appears very frequently in one class in the dataset.

We then found the 100 words with the highest z-scores according to the word's class distribution for words with more than 20 appearances. We then take those words and pass them through the model to obtain the model's bias for the given word with the procedure mentioned above. We then measure total accuracy and accuracy based on class.

2.1 Dataset Artifacts

With this data we were able to find words that had a higher class prediction than they typically should. We then tested the model and tracked the accuracy for test samples that contained artifact and non-artifact words.

Class	Total	Artifact	Non-Artifact
Entailment	90.4	92.2	89.6
Neutral	81.8	83.9	88.1
Contradiction	91.2	93.0	91.1

Table 1: Accuracy across data subsets

Next, we used this data to create a class of errors the model made where either the premise or the hypothesis contained words with high z-scores or artifacts in the training data. These words correspond to the colored points above the line in figure 1. Some examples of these words include 'better', 'will', and 'beds'. Three examples are provided below along with the model prediction and output probabilities:

"premise": "The food needs to be eaten soon."
"hypothesis": "The food will expire fast."
Model prediction: Contradiction
(34.67, 33.87, 31.46)

"premise": "A man with a cane walks away from a food cart."
"hypothesis": "The cane helps the man walk better."
Model prediction: Neutral
(36.13, 59.42, 4.45)

"premise": "Two men are in a room with four visible beds."
"hypothesis": "Two men showing their bedroom to photographer."
Model prediction: Neutral
(2.21, 96.07, 1.71)

2.2 Improvement

Using insights gained from part 1 of the project we decided to take a look at the dataset cartography approach from Swayamdipta et al., 2020 to imporove performance. We were able to generate all the data necessary independently and only use their code in the form of a single function to generate the plot of the the dataset. We were able to find hard to learn and ambiguous samples, collect them, and train our model in various ways to seek improvements.

Because there seemed to be little overlap between the hard to learn and the data artifact sets, we decided to stick to replicating the dataset cartography paper for part 2. With that being said, we followed a very similar procedure to that in Swayamdipta et al., 2020. We trained the model and tracked various metrics across each training epoch.

As in the paper, we tracked correctness, variability, and confidence from epoch to epoch. Correctness was simply the accuracy of each prediction across epochs, confidence was the models average predicted probability of the gold label, and variability was the standard deviation across epochs. We did this for 5 epochs until the model reached 88.5% accuracy on the validation set. Then we split the training data into a hard to learn subset which included the hard to learn as well as the ambiguous samples. These were samples that had a confidence of ¡ 0.5, variability of ¿ 0.35, or accuracy of ¡ 0.6. Then we training the model on 3 datasets, a hard training set that was inspired by a boosting based approach which simply doubled the amount of hard to learn problems,

the standard dataset left as a comparison, and as a point of experimentation we also tested training the model on only the only hard dataset to measure performance on the only hard dataset. This was inspired by an intuitive idea that you should train what you're weakest at to improve the most.

Following this we tested the performance of these models against the only hard and a test set to measure how well the models generalized. Additionally, using the authors' code to generate the dataset map we were able to generate a map for the Electra model on the SNLI dataset (See Figure 2).

3 Results

The largest improvements that we saw was from training on the hard dataset which included more hard to learn samples. We saw a 0.1% increase in performance on the test set as compared to training on the normal dataset.

Model	Only Hard	Test
Base	21.2	88.0
Hard Training	**39.2**	**88.4**
Only Hard Training	62.3*	13.2
Full Dataset	30.8	88.3

Table 2: Accuracy across data subsets after different training methods
* This model was trained on this dataset

4 Analysis

Using some information from the dataset artifact analysis we were able to determine that the model still generally struggles to correctly classify neutral and contradictory samples. From analyzing the hard only dataset, we were able to calculate the number of each class:

label	Entailment	Contradiction	Neutral
Count	17,966	31,309	17,478

Table 3: Counts of each label in the only hard dataset.

In contrast to the results in the paper, we actually saw a very slight improvement in training on the hard dataset vs. training on the standard dataset. This might be due to differences in model selection. The paper states that they trained their ALBERT model for 5 epochs and were able to achieve 93.1% accuracy on the validation set when training on the whole dataset, and 92.6% when training on a random mix of hard to learn and ambiguous samples with the full dataset. On our end, we were only able to achieve a max test set accuracy of 88.4.

We were able to reduce the errors of the hard to train class which mostly consisted of contradictions. This led to slightly better performance as mentioned previously. However, when tested on some sample inputs from the dataset artifact analysis, the model became more confident in it's wrong answers. For example, below are the predictions of the same samples from before but this time with our best model and our model trained on the normal dataset:

> "premise": "The food needs to be eaten soon."
> "hypothesis": "The food will expire fast."
> Before: (34.67, 33.87, 31.46)
> Hard data: (45.39, 28.63, 25.97)
> Normal data: (26.38, 35.06, 38.55)
>
> "premise": "A man with a cane walks away from a food cart."
> "hypothesis": "The cane helps the man walk better."
> Before: (36.13, 59.42, 4.45)
> Hard data: (45.39, 28.63, 25.97)
> Normal data: (26.38, 35.06, 38.55)
>
> "premise": "Two men are in a room with four visible beds."
> "hypothesis": "Two men showing their bedroom to photographer."
> Before: (2.21, 96.07, 1.71)
> Hard data: (45.39, 28.63, 25.97)
> Normal data: (26.38, 35.06, 38.55)

Strangely, the models don't seem to be learning in the same direction. The hard data model learns more of the artifact while the normal data doesn't in the first one for example. We hypothesize that this is because the hard data contains more artifacts than the rest of the dataset, ergo when we train the model on a dataset with more artifacts, it will have more spurrious correlations.

Additionally, in the dataset artifact collection we ran into one problem with the approach. This problem was that testing was difficult since the validation and test sets contained very few occurrences of artifact words to make any conclusive statements on model performance. Typically these words only appeared a handful of times if at all. This made it more difficult to tell if the model was unlearning these correlations in testing.

5 Conclusion

Using Dataset Cartography and investigating dataset artifacts we were able to improve our model's performance by very slightly on the test set. We achieved this by training our model on a dataset that consisted of a higher concentration of hard to learn samples. This, however, did seem to an increase in the number of spurious correlations learned form the data. We experimented and compared the performance of training on only the hard subset of the training set, Our result was contrary to what Swayamdipta et al.2020 found who saw a slight decrease in performance in their ALBERT model on the SNLI dataset.

Acknowledgments

We'd like to thank Dr. Durrett and the TA's for always being helpful, informative, and facilitating a fantastic course!

References

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 9275–9293, Online. Association for Computational Linguistics.

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. 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.

Lipton 2017 Zachary C. Lipton 2017. The Mythos of Model Interpretability. arXiv:1606.03490 [cs.LG]

Clark et al.2020 Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre- training Text Encoders as Discriminators Rather Than Generators. In Proceedings of the International Conference on Learning Representations (ICLR).