Oracle: Data Augmentation for Improved Generalizability of Natural Language Processing Models

Hee Hwang and **Sudarshan Raghavan**
College of Information and Computer Sciences
University of Massachusetts Amherst

Abstract

We present a rapid (minimal human-in-the-loop supervision) data augmentation framework that improves the performance of natural language processing models in low resource settings. To augment data for a particular downstream task, we use DepCC, A Dependency-Parsed Text Corpus from Common Crawl. First, we filter the Common Crawl file using queries extracted from the downstream task's training set and retrieve relevant domain data using the BM25 retrieval algorithm. After getting the relevant data, we obtain dense embeddings using BERT. We apply K-nearest-neighbors to these dense embeddings to retrieve sorted candidate augmentation data for each query. Lastly, we label the candidate augmentation data using one strategy: a simple label propagation technique of assigning the same label as the query. To examine the effectiveness of this framework, we perform augmentation experiments on three downstream classification tasks from three domain: computer science, news, and BioMed. We find in our experiments that our data augmentation framework improves held out test performance.

1 Introduction

State-of-the-art natural language processing models despite proving superhuman performance on benchmark datasets (Devlin et al., 2019), are far from true natural language understanding. These models' brittleness can be demonstrated in numerous ways: by utilizing simple rules, we can create examples that can cause trained models to fail and methods that exploit model confidence can be used to generate nonsensical adversaries (Ribeiro et al., 2018). Modern techniques, when combines with manual efforts, have been used to generate examples on which production models fail (Ribeiro et al., 2020).

Such issues are more pronounced in cases when the training data is scarce. Creating small training sets is a common practice when developing domain-specific models such as building a business grade conversational system (Coucke et al., 2018). In such scenarios, the developers have to construct their own training sets which is costly and can introduce undesired artifacts.

One family of approaches used to tackle model brittleness (in scarce data scenarios) is data augmentation. Data augmentation refers to the mechanism of adding additional examples to the training set. When the model is trained on the augmented dataset, the hope is that the model is less prone to failure than had it been trained on the original dataset. Data augmentation techniques in NLP have enjoyed success but they are often specific to particular types of model failures and thus may require significant manual labour.

1.1 Task Description

Our goal in this project is to develop a new framework for rapid data augmentation that improves the performance of NLP models in low resource settings.

1.2 Motivation and Limitations of Existing Work

Although recent advances in natural language processing have led to the development of models that achieve superhuman performance on benchmark datasets, they are far from truly understanding language. For example, models are self-contradicting, fail in the presence of noise, and incorrectly produce different predictions on semantically similar inputs.

Data augmentation is one of a family of approaches that can be leveraged to address these issues. There have been prior studies done that applied data augmentation techniques to improve model robustness

(see the "Related Work" section below for more details). While augmenting a training set with examples of a specific type has been shown to improve model performance on examples of a particular kind, there is a dearth of understanding about what are the other downstream effects of data augmentation (whether and when can it improve performance on examples of different types) (Kaushik et al., 2020), (Min et al., 2020).
This observation motivates our work in better characterizing downstream effects of data augmentation for deep NLP models and developing new tools for rapid data augmentation to improve model robustness.

1.3 Technical Challenges that you faced in the project

Some of the challenges that we'd faced in this project include:

1. **Size of Common Crawl:** The main challenge faced in this project is the size of Common Crawl. Compressed, it encompasses 400 GB. Building a Solr index on all of Common Crawl took approximately 2 months. Even after indexing all of Common Crawl data, performing retrieval from all of the 365M web documents sometimes results in the Gypsum jobs getting killed due to exceeding disk quotas. Given this, we required to present augmentation results on a subsampled version of Common Crawl data comprising 35M web documents

2. **Making the framework end-to-end:** Trying to connect the pipelines for retrieval from Common Crawl and processing the retrieved Common Crawl data for augmentation in order to make the overall augmentation pipeline end-to-end was another technical challenge that was faced in this project.

1.4 Contributions of your project

Some of our contributions of our project are as follows:

1. Developed a general framework for rapid data augmentation that requires minimal human-in-the-loop supervision

2. Presented two augmentation strategies using our augmentation framework that results in improvement of held out test performance for a deep NLP model in low resource settings

2 Related Work

There have been some prior studies of data augmentation in NLP. Work by (Min et al., 2020) explored several methods to augment standard training sets with syntactically informative examples generated by applying syntactic transformations to sentences from the MNLI corpus. Their work reported improvements that generalized beyond a particular construction used for data augmentation which suggests that their augmentation technique caused the deep NLP model (BERT) to recruit abstract syntactic representations. Some work demonstrates that augmenting a training set with counterfactual examples improves classifier performance especially on counterfactual test examples. Neural language models have also been employed to create new training examples by replacing tokens in original training instances.

Work by (Li et al., 2020) tackled the problem of pretrained lanuage models that, though performs well on in-distribution test sets, their performance suffers on out-of-distribution test sets. Their method of data augmentation involves in linguistically informed syntactic transformations that automatically generate desired contrast sets. Applying this technique to augment training data improves models' performance on such contrast sets without affecting performance on the original data.

This work by (Kaushik et al., 2020) showed that classifiers that are trained on original data fails on counterfactually-revised data and vice versa. Also spurious correlations in these datasets are picked up by even linear models. However on augmenting the revised examples, these picked up correlations are broken up.

This work by (Kobayashi, 2018) presented a data augmentation technique for labeled sentences called *contextual augmentation*. A neural language model is employed that, for a given sentence, replaces words with other words, thus resulting in the generation of new training examples. This method of data augmentation improved the performance of CNN/RNN based classifiers.
Unlike these works, our proposed method of augmentation specifically considers the model's encoding of the training set.

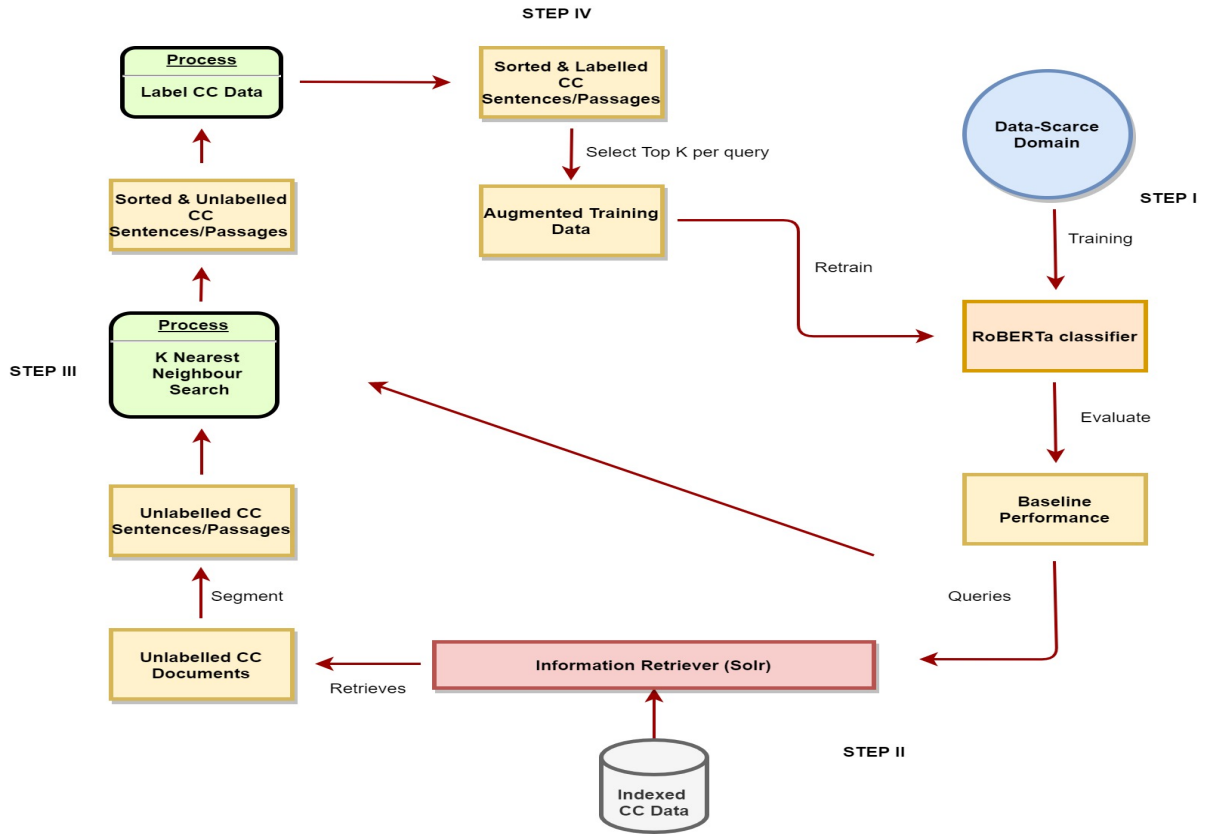


Figure 1: Augmentation Framework

3 Model

This section describes the details about the augmentation framework

3.1 Step I: Achieving baseline performance on target task

3.1.1 Data-Scarce Domain

We start off with the low resource (meaning that the training set contains a small set of examples) domain in which an NLP model will be trained to do a downstream task (in this case, classification).

3.1.2 RoBERTa classifier

An off-the-shelf deep NLP model such as RoBERTa would be finetuned for performing downstream classification task in the domain

3.1.3 Baseline Performance

After finetuning RoBERTa, it is then evaluated on a held out test set and this would serve as the baseline performance that has been achieved by RoBERTa in the target task

3.1.4 Queries

Based on the baseline performance achieved by RoBERTa in the target task, queries from the training set are extracted that would be used for retrieving relevant Common Crawl data

3.2 Step II: Retrieving from Common Crawl

3.2.1 Information Retriever

The queries that were generated in the previous step would then be fed into an Apache Solr based information retrieval system that, for each query, would return relevant web documents from Common Crawl. The retrieval system used in Solr is a BM25 scoring method. We have indexed all of the 365M documents in Common Crawl using Solr.

3.2.2 Unlabelled CC Documents

Solr returns a set of unlabelled CC Documents for each query.

3.3 Step III: Processing Retrieved Common Crawl Data

3.3.1 Unlabelled Common Crawl Sentences/Passages:

The retrieved Common Crawl web documents by Solr needs to segmented into sentences/passages. We use spaCy tool that does segmentation of these web documents into coherent sentences/passages.

The size of the sentences/passages required is decided by computing the average sentence/passage length as seen in the training data.

3.3.2 Process: K Nearest Neighbour Search

Here, we apply k nearest neighbour search that would sort the list of unlabelled Common Crawl sentences/passages by descending order of relevance for each query. We use dense embeddings from BERT of the query and its corresponding list of retrieved sentences/passages as input to the KNN algorithm and a Euclidean distance metric for achieving the ranking.

3.3.3 Sorted & Unlabelled Common Crawl Sentences/Passages

The KNN search algorithm applied in the previous step would return for each query, a list of Common Crawl sentences/passages that are sorted in descending order of relevance. However at this stage, we're still dealing with unannotated Common Crawl data.

3.4 Step IV: Labelling and Augmentation

3.4.1 Process: Label Common Crawl Data

Here we describe one unsupervised approach towards labelling Common Crawl Data:

1. Use the same ground truth label of the query (from the training set) as pseudo for its corresponding set of retrieved & sorted Common Crawl sentences/passages

3.4.2 Sorted & Labelled Common Crawl Sentences/Passages

Applying one of the two labelling strategies from the previous step results in, for each query, a list of sorted and labelled Common Crawl sentences/passages.

3.5 Augment Training Data

For each query, we select top K "least distant" Common Crawl sentences/passages and augment the target task training data with it

3.5.1 Retrain and Re-evaluate

After augmenting the training data the final step is to simply retrain the RoBERTa classifier on the augmented training data and re-evaluate its performance on the held out test set

3.6 Training

In terms of finetuning RoBERTa to perform downstream classification in the target domain, the hyperparameters we chose include:

1. Number of epochs: 10
2. Patience: 3
3. Batch Size: 16
4. Learning rate: 2e-5
5. Dropout: 0.1
6. Number of feedforward layers: 1
7. Feedforward non-linearity: tanh
8. Number of classification layers: 1

4 Experiments

4.1 Datasets

There are two categories of datasets that we used in this project.

4.1.1 Datasets for Classification Tasks

From the CS domain, we chose ACL-ARC (Jurgens et al., 2018) dataset for a citation intent classification task, from the BioMed domain, we chose RCT (Dernoncourt and Lee, 2017) for a sentence classification task in abstracts of randomized controlled trials and from the News domain, we chose HYPERPARTISAN (Kiesel et al., 2019) for a hyperpartisan news detection task. The reason for selecting each of these tasks is that we wanted to demonstrate how augmentation from Common Crawl can help augment training sets from different domains due to the diversity of data one would expect from Common Crawl.
A point to note: the citation intent classification and hyperpartisan news detection are by nature tasks in low resource settings (as seen in their size of training sets). However, this wasn't the case for the RCT dataset. Hence for the experiments, we subsampled RCT to 500 training examples and called it "RCT-sample."

4.1.2 Dataset for Common Crawl

For augmentation, we used DEPCC, a sanitized version of Common Crawl. It consists of 365 million documents that in total consist of 14.3 billion sentences.

Domain	Task	Label Type	Train	Dev	Test	Total
BioMed	RCT	abstract sent. roles	180040	30212	30135	240387
CS	ACL-ARC	citation intent	1688	114	139	1941
News	Hyperpartisan	partisanship	515	63	64	642

Table 1: Details on the domain specific datasets for classification tasks

4.2 Augmentation Strategy

Here's one data augmentation strategy that we experimented with (regarding the pipeline as described in Section 3)

1. Strategy 1 (Abbreviated as "S1"): The queries to the Solr retrieval engine being all the training examples from the training data of the target task. The labeling strategy applied here is essentially applying the same ground truth label as the query for generating pseudo labels its corresponding set of retrieved Common Crawl sentences/passages.

4.3 Massive Augmentation

Using Strategy 1, we augment the data based on the maximum distance between the dense embedding of the original query and retrieved passages. In short, we add every passage whose distance is less than a threshold. Figure 2, 3, and 4 shows the size of the training data after augmentation. The x-axis corresponds to the maximum distances, and the y-axis the size of the training data. We trained each model *five* times and calculated average and standard deviation.

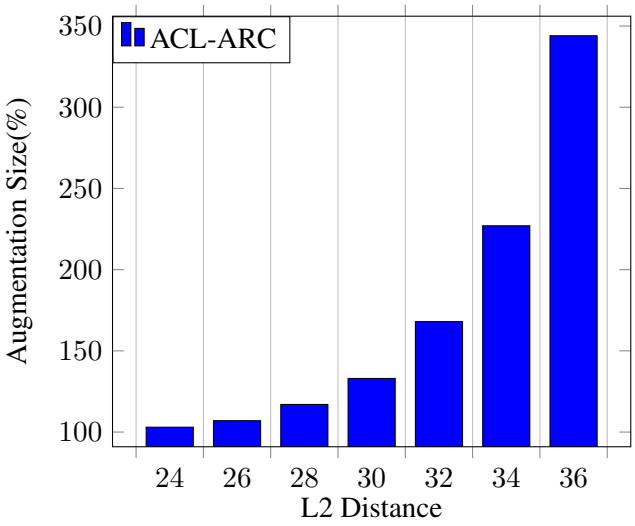


Figure 2: ACL-ARC Augmentation Size

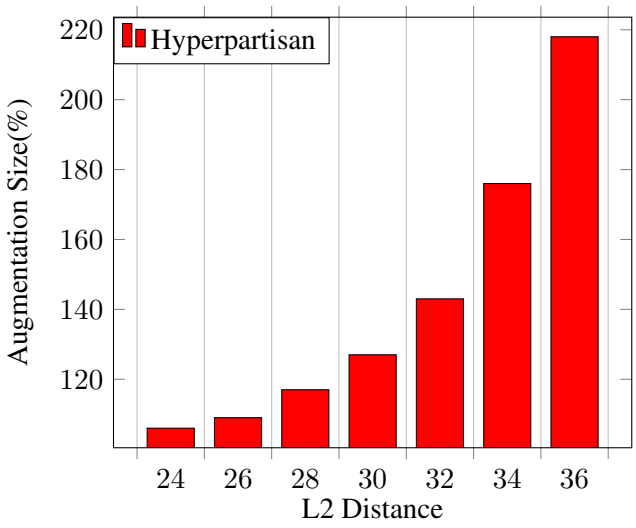


Figure 3: Hyperpartisan Augmentation Size

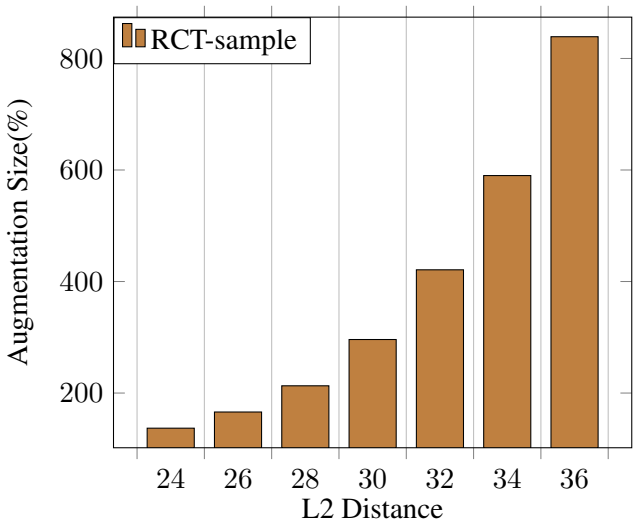


Figure 4: RCT$_{sample}$ Augmentation Size

4.4 Fixed-size Augmentation

We also defined another augmentation using a small amount of retrieved data. We fix the number of augmenting data. For example, if we have 1000 training examples, we pick 30 examples from the retrieved data and add them to the training set. Again, we trained and tested each model *five* times.

Figure 5: Massive Augmentation

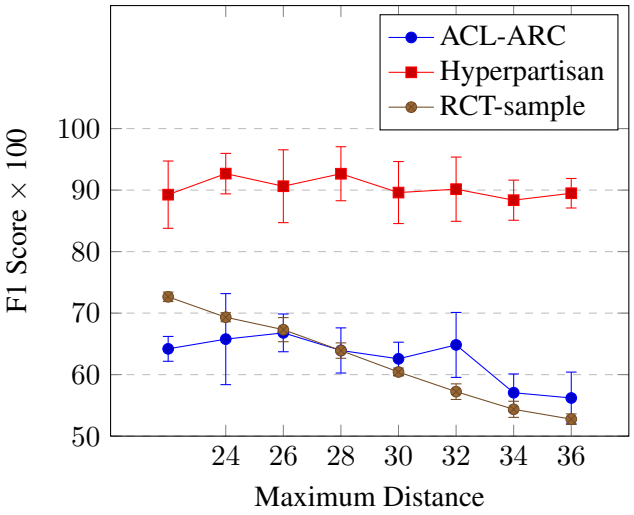
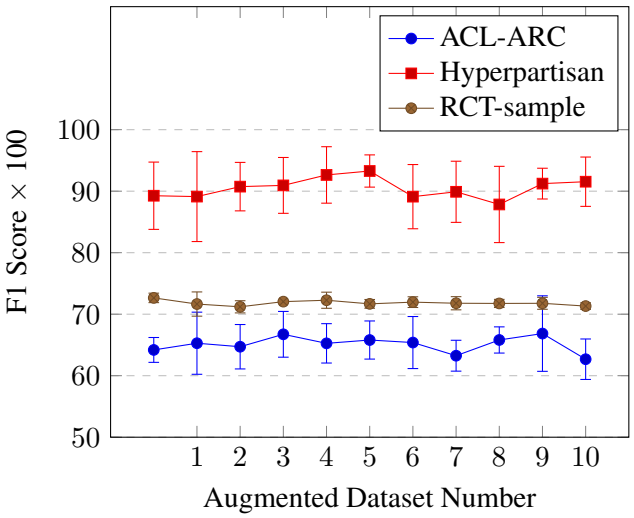


Figure 6: Fixed-size Augmentation



Average F1 Score (Standard Devation)			
Dist	ACL	Hyper	RCT$_{sample}$
Base	64.19(± 2.02)	89.26(± 5.47)	**72.65(± 0.77)**
24	65.77(± 7.40)	**92.68(± 3.29)**	69.33(± 0.73)
26	**66.79(± 3.07)**	90.63(± 5.92)	67.31(± 1.96)
28	63.93(± 3.67)	92.66(± 4.39)	63.91(± 1.25)
30	62.57(± 2.70)	89.60(± 5.04)	60.43(± 0.64)
32	64.83(± 5.28)	90.15(± 5.22)	57.23(± 1.27)
34	57.06(± 3.05)	88.36(± 3.26)	54.36(± 1.31)
36	56.20(± 4.21)	89.49(± 2.40)	52.76(± 0.85)

Table 2: Massive Augmentation

Average F1 Score (Standard Devation)			
Data #	ACL	Hyper	RCT$_{sample}$
Base	64.19(± 2.02)	89.26(± 5.47)	**72.65(± 0.77)**
1	65.28(± 5.05)	89.12(± 7.31)	71.65(± 1.97)
2	64.71(± 3.61)	90.74(± 3.94)	71.20(± 0.99)
3	66.73(± 3.72)	90.94(± 4.54)	72.04(± 0.20)
4	65.26(± 3.20)	92.64(± 4.59)	72.27(± 1.31)
5	65.80(± 3.10)	**93.28(± 2.62)**	71.68(± 0.71)
6	65.39(± 4.23)	89.11(± 5.22)	71.96(± 0.87)
7	63.25(± 2.51)	89.90(± 4.97)	71.78(± 1.07)
8	65.81(± 2.13)	87.84(± 6.20)	71.75(± 0.66)
9	**66.85(± 6.14)**	91.23(± 2.50)	71.77(± 0.97)
10	62.68(± 3.29)	91.54(± 4.01)	71.32(± 0.54)

Table 3: Fixed-size Augmentation

4.5 Main Results

1. Massive Augmentation
   Figure 5 and Table 2 shows that massive augmentation improves performance on ACL-ARC and Hyperpartisan at the beginning. F1 score of ACL-ARC and Hyperpartisan achieve 66.79% and 92.68%, respectively. However, the performance deteriorates as we add more data.

2. Fixed-size Augmentation
   As in figure 6 and Table 3, ACL-ARC dataset achieves 66.85% when augmenting the 9th dataset. For Hyperpartisan, adding 5th dataset performs the best(93.28%). Again, augmenting RCT$_{sample}$ did not improve F1 scores.

4.6 Analysis

1. Massive Augmentation
   Figure 5 and Table 2 show the F1 scores of three datasets after augmenting data with the maximum distance metric. The F1 score of ACL-ARC and Hyperpatsan News improves up to a certain point and deteriorates after. For RCT$_{sample}$ dataset, it keeps decreasing. We believe that the common crawl dataset may not be a good choice for augmenting BioMed data.

2. Fixed-size Augmentation
   We've shown that massive augmentation may not be beneficial for improving the F1 score. Therefore, we hypothesized that a small num-

ber of data could improve the F1 score. For Figure 6 and Table 3, we only augment only 3% of the training set. The result clearly shows that the distance to the original query is not the defining factor for achieving the best F1 score.

3. RCT$_{sample}$
   A reason why S1 didn't outperform the baseline in the RCT is probably that the 35M sampled Common Crawl (which was used for augmentation) didn't contain much data from the BioMed domain.

4.7 Qualitative Analysis

We examine a closest and farthest passage from a given query from citation intent classification task. It is evident that the closest passage has a similar structure, including topic, author, and publishing year as the query. Assigning the same class("background") contributes to the model's high performance with this augmented data. However, the farthest passage is more like a product description or an advertisement.

- Query

  > Thus , over the past few years , along with advances in the use of learning and statistical methods for acquisition of full parsers (Collins , 1997 ; Charniak , 1997a ; Charniak , 1997b ; Ratnaparkhi , 1997)...

- Closest passage

  > It is local , yet can handle also compositional structures . ... full parse of free - text sentences (e.g. , Bod (1992) , Magerman (1995) , Collins (1997) , Ratnaparkhi (1997) , and Sekine (1998))

- Farthest passage

  > Festival Speech Synthesis System : From the Centre for Speech Technology Research at the University of Edinburgh .Festival offers a general framework for building speech synthesis systems ...

5 Conclusion and Future Work

Some of the future work that we're currently pursuing for this project include:

1. Achieving diversity in data augmentation: Perturbation of queries using a paraphrasing model could be a technique to achieve diversity in augmentation

2. Performing retrieval from all of Common Crawl Data (365M Web documents)

3. We are currently experimenting with another unsupervised approach towards labelling unannotated Common Crawl data - using the baseline RoBERTa classifier to generate pseudo labels (inspired by Self-Training techniques utilized by (Xie et al., 2020)

Acknowledgements

We wish to thank Kalpesh Krishna, the PhD mentor for this project for providing incredible insights through the course of this project. We would also like to express our gratitude towards our industry mentors from Oracle: Ari Kobren, Swetasudha Panda, Michael Wick and Naveen Jafer Nizar for also providing solid advice and directions throughout the course of this project.

References

Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, Maël Primet, and Joseph Dureau. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406.

Divyansh Kaushik, Eduard Hovy, and Zachary C. Lipton. 2020. Learning the difference that makes a difference with counterfactually-augmented data.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on*

Semantic Evaluation, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Sosuke Kobayashi. 2018. Contextual augmentation: Data augmentation by words with paradigmatic relations. 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 452–457, New Orleans, Louisiana. Association for Computational Linguistics.

Chuanrong Li, Lin Shengshuo, Zeyu Liu, Xinyi Wu, Xuhui Zhou, and Shane Steinert-Threlkeld. 2020. Linguistically-informed transformations (LIT): A method for automatically generating contrast sets. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 126–135, Online. Association for Computational Linguistics.

Junghyun Min, R. Thomas McCoy, Dipanjan Das, Emily Pitler, and Tal Linzen. 2020. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2339–2352, Online. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. 2020. Self-training with noisy student improves imagenet classification.