Sigmoid Based Soft Loss for F1 Optimization

Stanford CS224N Default (RobustQA) Project

Bhavik Shah
Institute for Computational and Mathematical Engineering
Stanford University
bas336@stanford.edu

Sudeep Narala
Electrical Engineering
Stanford University
sudeepn@stanford.edu

Abstract

When building robust machine learning models, it is often desirable to train models that are able to generalize to data that is not necessarily from the same distribution, but still of a similar formff. Specifically in the field of Question Answering, we want to be able to train a DistilBERT model on some larger dataset, and then still have it perform well on some more niche out-of-domain dataset. In this paper, we apply a number of techniques that can help us solve this problem. We utilize model-agnostic meta learning (MAML), a method useful for fine-tuning models in a few-shot learning paradigm. Moreover, we apply data augmentation (masking), as well as architectural shifts such as adding more transformer layers and modifying the loss function. We apply these methods because they also provide some interpretability when looking at their performance overall. With respect to performance, the meta-learning methods tended to perform the most poor out of the bunch when compared to the baseline; however, our model that was trained with an auxiliary loss function performed very well, achieving a 72.06 in-domain val F1 (1.92% improvement), 52.28 out-of-domain val F1 (4.8% improvement) and test F1 of 62.23 which put us at 4th on the leaderboard (top 10%).

1 Key Information to include

TA mentor: Fenglu, External collaborators : No, External mentor : No, Sharing project : No

2 Introduction

With the rise in availability of high quality language data, as well as the constant improvements made in machine learning models, large scale question-answering problems have been pushed further and further into the forefront of the natural language processing field. Now, extremely expressive transformer-based models such as BERT [1] are considered state of the art in these types of tasks. However, we do notice that these models require large quantities of data in order to be effectively trained in some question-answering task. Moreover, we notice that these models often struggle to generalize to datasets of questions that are out-of-domain, which often have smaller quantities of - and more niche - information. As such, we explore techniques in this paper that can be useful for the adaptation of larger models (here, we specifically use DistilBERT [2], a smaller version of BERT) to these smaller out-of-domain question answering tasks.

One field that has become increasingly popular in reinforcement learning and classification domains in recent years is meta-learning, which is the process of "learning how to learn". More concretely, we are able to learn more general purpose parameters such that we can encourage the model to learn how to quickly tune itself to a new, unseen task. These methods can be particularly useful in few-shot learning situations, as a model can quickly learn how to adapt itself when given only a few examples. As such, it is particularly suited to be utilized for NLP tasks as well when we have these smaller out-of-domain sets of data.

Additionally, we try other methods for encouraging the model to perform well on the out-of-domain sets. Particularly, we try forms of data augmentation, where we modify the out-of-domain datasets with forms of masking, in an en effort to make more data (these datasets can oftentimes be quite small), as well as create more challenging examples for the model to learn.

Moreover, we try methods that involve modifying the final layers and/or loss function of the model, such that it leads to a more flexible model when faced with new information. For example, we replace the final transformer with an architecture that allows it to choose weights that operate best at a given time for that example (ie, there are multiple transformers in parallel that might work differently on different domains). Moreover, we experiment with ideas that allow for a softer loss function, allowing the model to differentiate between being close to the correct answer.

3 Related Work

The field of meta-learning has evolved greatly over the past couple of years; one of the pivotal paper being that which introduced model-agnostic meta learning [3]. While the algorithm is discussed in detail in the following section, this paper introduces a framework where the any model can learn general parameters that are suited to learn some specific, new task with only a few support examples. While its results shown in the paper are quite good, it has a some key drawbacks – for example, it is highly sensitive to the support examples given to the model, as well as the distribution of tasks provided to the model. Another key limitation is that the implementation of MAML involves the computation of second-order gradients, which is often too resource-taxing for practical use. One way to alleviate this issue is introduced in [4], which empirically shows that in many cases only using meta-learning on some smaller subset of the model (given we are fine-tuning) can work almost as well as doing so on the whole model. This allows for significantly reduced computational cost. Moreover, many first order approximations also exist, such as Reptile, introduced in [5]. We leverage these paper to fine-tune our model, which is an interesting approach as the applications of question-answering and MAML is an under-explored field.

Moreover, we also explore soft-labeling ([6]). Since many BERT models rely on cross entropy loss, it can often lead to high loss terms even if the model was close in predicting the start and end tokens of the answer. Soft labeling is a popular technique for classification, especially when more than one label would make a reasonable guess. Soft labeling is often not used in NLP tasks, so we felt that by using these techniques for our purposes, we can build a model that is optimized for a good F1 score rather than Exact Match (EM). However, the clear drawback here is that the model can often be encouraged to only have to make "good" predictions, since loss is reasonably low if it is close to the correct answer. As such, it is important to find a balance with how lax to be when looking at the soft labeling; this is something we explore in our paper.

4 Approach(es)

4.1 Meta-Learning

We started with an implementation of Model-Agnostic Meta Learning (MAML) [3], which we have implemented (and its associated data loaders, which were modified from the provided ones) ourselves. We utilize this method to fine-tune the DistilBERT model on the out-of-domain datasets. The MAML model is described in figure 1 (figure also from [3]). It consists of an outer step and an inner step. In the outer step we sample a batch of tasks, where a task is some question domain that we are fine-tuning on. Each task is the comprised of K questions in the support set, and then Q questions in the query set. Then, for each task, (described lines 4-6 in the algorithm), we adapt the parameters of the model according to the performance on the support examples to get some new adapted parameters θ_i. We then look at the performance of the adapted parameters on the query examples, and record that loss. We then perform another gradient update step using the gradient of the loss on the query examples, based on the original un-adapted parameters. Thus, we see that the final goal is such that we learn some parameters θ such that we are able to quickly adapt to some new task, given a small amount of examples.

Note that our implementation uses an Adam optimizer to make the update described on line 8, not the vanilla SGD that is implied. Moreover, during the actual MAML fine-tuning procedure, we only meta-learn on parameters θ that are those of the final linear layer. This approach is described in [4],

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Figure 1: MAML Algorithm

and is shown to be almost as performant as vanilla MAML, but with significantly less cost (as the gradient computation is not as large). Also note that for parts of the MAML implementation, we relied on the CS330 HW2 (which Bhavik took and completed this assignment for) implementation of MAML.

4.2 Parallel MAML

We try an small ensemble-like technique when using MAML, where we meta-learn on some subset of layers of the DistilBERT model. When it comes time to evaluate the performance of the model, we use the original weights of the baseline model to get one set of predictions, as well as the meta-learned parameters to get another set. We then look at the confidence scores of each of the two models, and predict whichever one has a higher score.

4.2.1 Reptile

We note that the implementation of MAML requires the computation of second order gradients, specifically when computing the update on line 8 of Figure 1 . As such, MAML requires an extremely high computational cost, which is an issue that often renders second-order methods impractical. As such, that same update step often finds itself replaced with first-order approximations. One popular method is Reptile [5], which replaces the update with

$$\theta \leftarrow \theta + \frac{\epsilon}{n} \sum_{T_i \in B} (\theta_i' - \theta)$$

, where ϵ is some hyperparameter akin to a learning rate, and this specific update is batched over a series of asks (although the update can also be done after each task individually).

4.3 Data Augmentation

Since we had very little out of domain data training data to work with, we tried to augment it using masking.

4.3.1 Masking Context

The first thing we tried was masking the context in a paragraph. In this approach, we ensured that the question and answer tokens weren't being masked. We used a base DistilbertForMaskedLM from the huggingface library in order to infer the masked tokens.

Figure 2: Masking the Context

4.3.2 Masking Answers

After considering the previous approach, we thought that masking the answer would be an even better way to ensure that the model is robust. Typically, the actual content in the answer shouldn't affect the answer. Instead the model should ideally rely almost exclusively on the context. Here is a simple example to illustrate.

Eg: Question: How much does a bar of soap cost?
Context 1: A bar of soap costs **about the same as a candy bar**.
Context 2: A bar of soap costs **$5**.

In both these situations, the answer remains in the same place relative to the context. With this intuition, we sought to augment data by masking answers.

Figure 3: Masking the Answer

4.4 Transformer Level Attention

There were 2 goals with this architecture. 1) Make each model specialize in the different types of logic needed for question answering and 2) Provide some interpretability to the model. We do this by attempting to make inputs choose between the different transformers, as shown in 4.

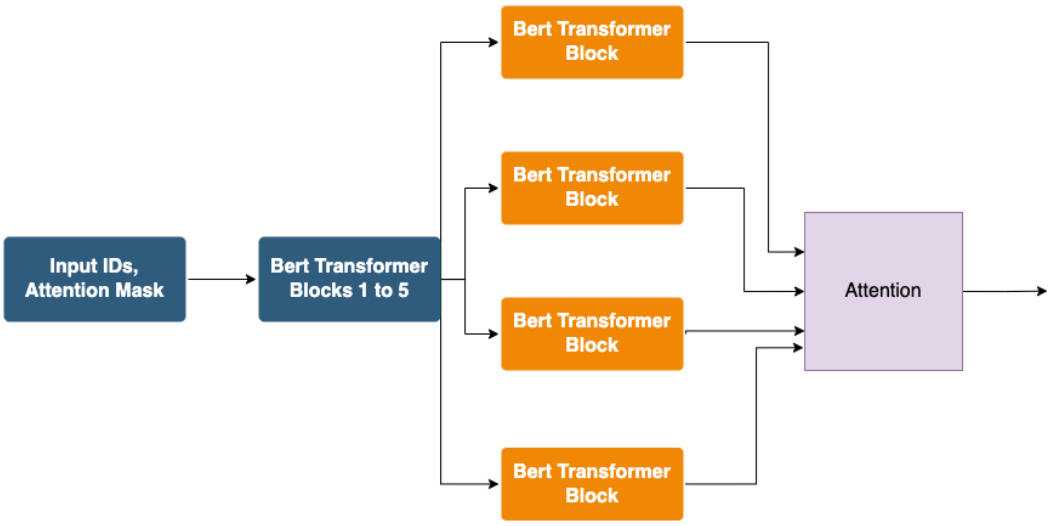

Figure 4: Transformer Level Attention Model

The attention for each of the blocks is based on the value of the [CLS] token.

The goal was to use the score vectors corresponding to an input (in the case of 4, 4D vectors) in order to cluster the questions. I.e. if 2 vectors had similar scores, we would expect them to correspond to inputs with some similarities.

4.5 Auxiliary Smooth Loss

One big thing we noticed is that the default loss function for DistilBertForQuestionAnswering is a cross-entropy loss who's goal is to maximize EM, and which maximizes F1 more implicitly than explicitly. Since our goal is to improve the F1 score, which tends to be a more robust capture of the usefulness of a model, we tried to add in an auxiliary loss term that penalized the model less if it is in

the right area of the answer, and more if it is completely off. This came in the form of a smooth loss coefficient surrounding the correct index of a start/end position. The details are shown below.

Suppose i is the location of the ground truth position (for start or end)

$$loss = -log(y[i]) - \sum_{n=1}^{max\_query\_len} f(n) \cdot log(1 - y[n])$$

where

$$f(x) = 2 \cdot |\sigma(\frac{x - i}{\gamma}) - 0.5|$$

The hyperparameter γ is the smoothing factor and y contains the model probability predictions for each location.
The first term in the loss is the regular cross-entropy loss. The second term penalizes the model less for predicting locations that are close to the ground truth. One improvement that can be made on this loss is to make γ based on the specific input's answer length so the penalty can be assessed accordingly. I.e. if an answer is longer, make the function smoother and more lenient for longer lengths from the ground truth.

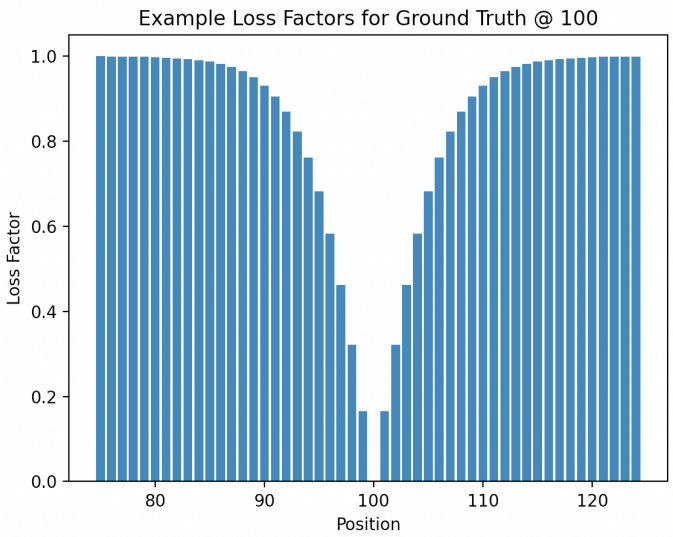

Figure 5: Loss Factors for $i = 100$, $\gamma = 3$

Some more intuition to this approach is that with a loss that isn't as stringent on answers that are less wrong, the model might be able to generalize better to datasets it has never seen before. We are incentivizing the model to get close to the correct answer and don't penalize it as much when it is not the exact right answer, which might make its embeddings more robust to domain shifts.

5 Experiments

5.1 Data

We use the datasets provided for the RobustQA track: SQuAD , NewsQA and Natural Questions for training the initial baseline DistilBERT model. We then use the out-of-domain datasets: RACE [7], DuoRC[8], and RelationExtraction [9] to finetune the model using the various methods previously described.

5.2 Evaluation method

The evaluation of each method is done by computing the Exact Match (EM) and F1 scores on both the out-of-domain validation and out-of-domain test sets.

5.3 Experimental details

5.4 Meta-Learning Models

For MAML models were both implemented with an outer learning rate of $3e-5$, and an inner learning rate of 0.3. We gave 10 support examples and 5 query examples for each task iteration, and we had three tasks for meta-training, where each task corresponded to one of the in-domain datasets. Finally, we fine-tuned the meta-learned parameters on the out of domain data before evaluating our model.

5.4.1 Transformer Level Attention

We froze the first 4 transformer blocks and the word embeddings for training the transformer level attention model to speed up training time.

5.4.2 Auxiliary Smooth Loss

We didn't have as much time to test this method, so we were able to only test it with $\gamma = 4$ for the smooth factor.

5.5 Results

Following are our results for the out-of-domain validation and test.

Model Type	F1	EM
Baseline	49.881	34.56
Vanilla-MAML	45.021	28.092
ParallelMeta	47.858	31.675
Baseline + OOD Finetune	51.24	**35.86**
Baseline + Answer_Masking	50.71	35.60
Transformer Level Attention	49.56	34.29
Transformer Level Attention + OOD Finetune	50.01	35.34
Transformer Level Attention + Context_Masking(0.15)	49.22	34.29
Transformer Level Attention + Context_Masking(0.25)	49.00	33.77
Transformer Level Attention + Answer_Masking	51.32	35.34
Auxiliary Smooth Loss	**52.28**	35.08

Table 1: Model Results (Dev Set)

Model Type	F1	EM
Auxiliary Smooth Loss	**62.225**	**43.028**
Baseline + Answer_Masking	57.846	40.940
Transformer Level Attention + Answer_Masking	57.790	39.885

Table 2: Model Results (Test Set)

Note that the results of Reptile are not reported as they performed quite worse than the vanilla second order MAML, as we did not have time to sufficiently test its hyperparameters.

As mentioned in section 4.2, we expected these results because we thought the auxiliary smooth loss would do a better job generalizing to unseen datasets. The test set, which is far more vast than the validation set agreed with this analysis.

6 Analysis

6.1 Meta-Learning

We see that for the MAML models, performance especially lacks when compared to the baseline. In general, we believe that the task distribution of the out-of-domain question answering problem

is not as clear-cut as more canonical meta-learning problems (ex. Omniglot). As such, it is hard to fine-tune the model in such a way that it is prepared for the few-shot scenario of the out-of-domain questions. We do however see a performance gain on the Parallel MAML model when compared to the normal MAML model. This might be the case just because the parallel model has the option of defaulting to the baseline when it is not confident in the meta-learned predictions.

6.2 Transformer Level Attention

Although the hope was to obtain vectors corresponding to different inputs that could be used for input clustering, our results didn't align with our expectations. There was little variation between inputs and even when there was, we couldn't see any clear clusters forming, as shown in the PCA plot.

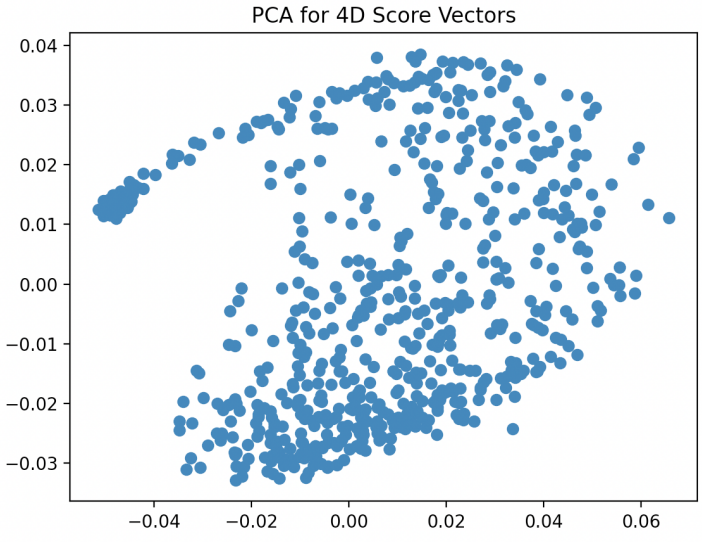

Figure 6: PCA for 4 Transformer Score Vectors after 2 epochs

6.3 Auxiliary Smooth Loss

This method yielded some very significant results compared to the other methods. We can see in 7 that the smooth loss model remains higher in F1 and lower in EM on the in-domain val dataset throughout the run duration. This makes sense for the in-domain dataset. Since there is a large abundance of data, the model trained on just the EM objective is able to outperform the smooth loss model in EM. However, this is no longer the case for the out-on-domain data. As discussed earlier, this may be because a model trained just on the EM objective isn't able to generalize as well to datasets it has never seen before due to the stringent loss, not rewarding the model for getting close.

6.4 Performance on Different Datasets

Model Type	F1	EM
Auxiliary Smooth Loss	**62.225**	**43.028**
Baseline + Answer_Masking	57.846	40.940
Transformer Level Attention + Answer_Masking	57.790	39.885

Table 3: Model Results (Test Set)

7 Conclusion

Overall, we conclude that MAML doesn't seem like a very principled way to approach the problem due to the task distributions not being quite clear, although perhaps other meta-learning paradigms

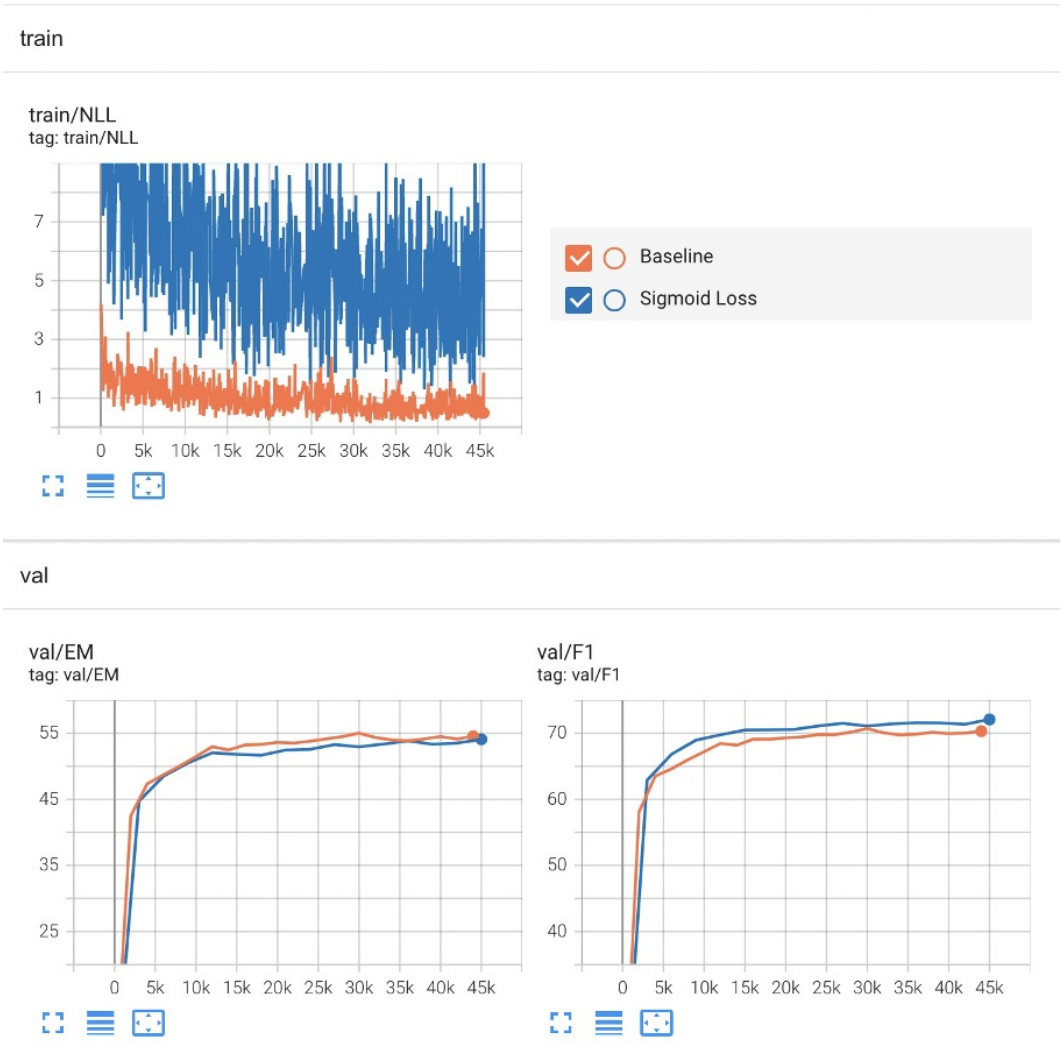

Figure 7: Results vs Baseline Model

might be useful, and might be interesting to explore in the future. Also, while our clustering method seems principled in approach, there leaves something to be desired in terms of its final interpretability and results (they were not as robust as we would have liked). However, our auxiliary loss performs the best by far and does quite well on the dev and test sets, suggesting that this type of penalty (being less harsh if we are close) is useful for the question answering problem. Due to the long training time of some of our models, we were unable to explore a larger amount of tuning. We would have also liked to do a little experimenting with slight modifications to our architectures.

For future work, we would like to combine a lot of our methods, since a lot of our approaches were general techniques. For example, data augumentation would have been nice to create more masks for use with MAML, and the sigmoid loss function could be applied to nearly any other approach as a supplement.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[2] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.

[3] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400, 2017.

[4] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. *CoRR*, abs/1909.09157, 2019.

[5] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *CoRR*, abs/1803.02999, 2018.

[6] Doyup Lee and Yeongjae Cheon. Soft labeling affects out-of-distribution detection of deep neural networks. *CoRR*, abs/2007.03212, 2020.

[7] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017.

[8] Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. Duorc: Towards complex language understanding with paraphrased reading comprehension. *CoRR*, abs/1804.07927, 2018.

[9] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. *CoRR*, abs/1706.04115, 2017.