Building a Robust QA System

Stanford CS224N Default Project

Amit Singh
SCPD Student
Stanford University
amitsing@stanford.edu

Mohd Zahaib Mateen
SCPD Student
Stanford University
zahaib@stanford.edu

Abstract

Question Answering has been a budding section of NLP research. Pre-trained neural models for QA Systems have shown impressive results while working with in-domain data. However, their robustness to generalize on out-of domain data has been an active area of research. With the baseline of a pre-trained DistiIBERT model, we plan to work on several techniques to improve the robustness of the QA system. The various techniques we plan to explore include longer fine-tuning with ood dataset, data augmentation using back-translation , domain adversarial modeling and hyper parameter tuning.

1 Key Information to include

- Mentor: Elaine Sui
- External Collaborators : N/A
- Sharing project: N/A

2 Introduction

Natural Language based Question Answering Systems as an application have diversified across a range of domains and tasks. Typically, these systems are trained on large, homogenous, use-case specific datasets; however it is then challenging to reuse the model in cases where the data changes drastically. The naïve next step is to retrain a new model on the new dataset. This approach is clearly sub-optimal and often impractical. The challenge we have attempted to overcome with this project is how to transfer the learning of a BERT based language model in a way that it adapts well to unseen datasets in the QA Domain. Or to phrase it differently, the model must be robust enough to generalize to unseen datasets. This has been the topic of active research as often these models tend to overfit to the datasets that they're pre-trained on. The idea is to have a QA system pretrained on a large dataset A. It is then finetuned on a new, much smaller dataset B using techniques that allow it to perform well on B leveraging the learning from being trained on A. It is challenging because while the intention is for the model to understand and learn patterns, it is counter-productive if it instead ends up memorizing specific questions and/or their answers.

To achieve this, we have explored techniques like Data Augmentation through Backtranslation, where we have back translated parts of the context to a set of "pivot" languages and then translated them back to English, a Domain Adversarial Model that penalizes the model if it learns domain specific encodings and hyper-parameter fine-tuning with the learning rate. While we will go into the details of these approaches, the general idea behind these approaches has been two-fold:

- To make the most of the data available in training the model
- To penalize the model if it tends to overfit on a specific domain

We use Exact Match and F1 scores as our evaluation metrics and have achieved significant improvement over the baseline, details of which we'll cover in upcoming sections.

3 Related Work

Our work towards improving the QA Model performance was inspired from a range of works undertaken by contemporary authors in this field. There were several techniques that, as we discovered, were used to deal with this task of building a Robust QA system. The idea was to understand the approaches that best suit our specific use-case. We attempted to explore Masked Level Modelling as a possible solution to begin with however we found it will not be the best representative of our QA dataset. Other Major Works that we studied for our problem include:

Lee et. al [1] has been our primary reference work as far as Domain Adversarial Training is concerned. In. this work, the authors make use of an adversarial training framework for domain generalization in context of Question Answering. They use a discriminator besides a BERT based QA Model. Adversarial training ensures that the QA model is penalized if it attempts to learn domain-specific features. A MRQA Shared Task is used for application and the design shows significantly better performance compared to the baseline model.

Longpre, Shayne, et al. [2] have demonstrated a different strategy in dealing with the problem of making a Robust QA System. They explore a negative sampling technique to facilitate transfer learning from large, pre-trained models across domain shifts. The work included backtranslation with both the query and the context. For generating context paraphrases, the approach uses SpaCy to segment each context into sentences. Then they implement back-translation on a sentence-by-sentence basis. The results were found to be particularly effective, even though the datasets used; such as SQuAD 2.0 included unanswerable questions. The approach achieved the second best EM and F1 cores in the MRQA leaderboard competition.

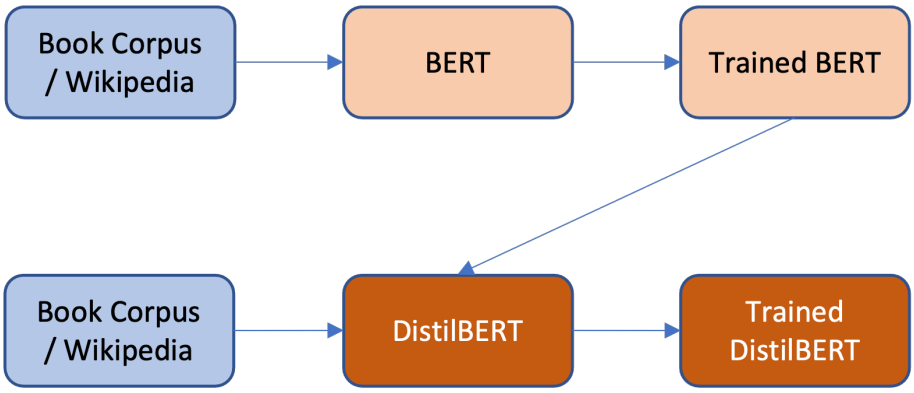
Other approaches for dealing with this problem include a Mixture-Of-Experts [3] (MoE) technique where k different models are trained along with a gating function that decides the importance of each model towards the final output. Or a task adaptive Pre-training [4] approach where the QA based loss is jointly optimized along with the original masked language modeling (MLM) loss over the inputs. Our model attempts to make the best of both of the first two approaches (Domain Adversarial Training and Data Augmentation) as we believe they tend to solve two different problems, both of which are critical to the problem at hand.

4 Approach

For purpose of setting up a baseline model, we finetune a DistilBERT [5] model. Given i and j are the gold start and end locations, our aim is to use the AdamW optimizer to minimize the loss given by the following equation:

We have worked towards improving the baseline performance through the following approaches:

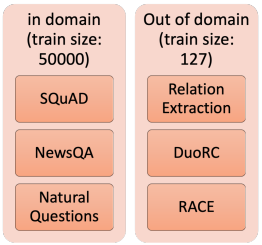
- Longer fine tuning with ood dataset (8 epochs)
- Data augmentation using Backtranslation [2]
- Hyperparameter-fine-tuning



5 Experiments

5.1 Data

For training the QA system, the in-domain datasets namely Natural Questions, NewsQA and SQuAD are used. They have 50,000 examples each for training. There are an additional 27,555 examples in total for the dev set.



For evaluation, three out-of-domain datasets namely RelationExtraction, DuoRC and RACE are used. The idea is to test the performance of our model on the out-of-domain test sets to measure the robustness of the model across domain shifts.

Sample Data Point:

Context:

On a brief trip back to London, earnest, bookish bacteriologist Walter Fane (Edward Norton) is dazzled by Kitty Garstin..... Kitty meets Charles Townsend (Liev Schreiber), a married British vice consul, and the two engage in a clandestine affair. When Walter discovers his wife's infidelity, he ... . .Kitty rejects his overtures and walks away. When her son asks who Townsend is, she replies "No one important, darling"

Question:

What is the name of the man Kitty has an affair with?

Answer:

"answer start": 555, "text": "Charles Townsend"

5.2 Evaluation method

We use F1 scores and EM (Exact Match) scores as the metrics for our evaluation as presented in the handout for RobustQA track[6].

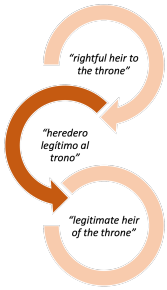
5.3 Experimental details

While fine-tuning the model on out-of-domain dataset we tried running the model with 2-3 epochs and pre-training for longer with 8,16 epochs. Through our experiments, we deduced that pre-training with 8 epochs gave the best F1 and EM scores reported under the results section. Below-mentioned are the details of each segment of our model. The default hyper-parameters used are as under:

batch size	16
eval every	2000
learning rate	3e-05
num of epochs	3
num of visuals	10
seed	42

5.3.1 Data Augmentation via Back Translation

The motivation here is to have a larger out-of-domain dataset for training; to have the model learn semantic similarities and discourage it from learning lexicographic similarities. We started off with the simplest approach of back-translating questions only, but that didn't work as there needs to be lexicographical similarity between q/a. We were missing out on effectively translating the q/a pairs and our performance went down. We then attempted to translate the entire context but that had situations of multiple answers getting missed. The quality of translation was also poor. We finally came up with a way to back translate parts of context that



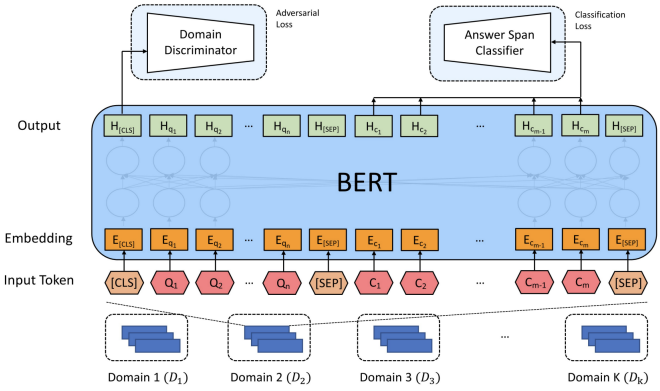
are unaffected by the q/a and that worked well. While this approach is inspired from works like (Longpre, Shayne, et al.) [2], the precise strategy was original to our work in this project. Here's how the algorithm works:

- Let normalized list of all answer start indexes be X; X = {X_1, X_2, ... X_z}
- Context is broken down on a sentence-by-sentence basis into k sentences (m_1, m_2, ... m_k).
- Start and end logits of each sentence (m_p) are calculated (m_p^s and m_p^e)
- For each sentence (m_p), if there exists x in X such that $m_p^s < x < m_p^e$, it is flagged as "false" for backtranslation
- Every other sentence is flagged as "true" for backtranslation
- Only the answers flagged "true" are backtranslated and the new, updated answer indexes are re-calculated and updated for all questions.

This strategy broke down the task into relatively smaller pieces and helped us with creating an augmented dataset. We iterated over a few languages including traditional Chinese, Spanish, French and Arabic and found that Spanish and French gave us the most relevant backtranslations. We used PyPI's BackTranslation API [7]

5.3.2 Domain Adversarial Training

Our work is inspired by the architecture given by works of Lee, Seanie, Donggyu Kim, and Jangwon Park [1]. We have adopted this architecture to our DistilBERT model where the domains are representative of the datasets they belong to.



Our intention here is to penalize the model for "memorizing" domain-specific encodings by training it in an adversarial manner. The Discriminator D classifies the joint embeddings for (q,c) into d domains . The general idea is to project (q,c) in an embedding space where the discriminator cannot categorize them based on their domains. Based on the ID, we added labels to the data points that were representative of their domains. The overall loss function that we were attempting to optimize is given below:

$$\mathcal{L}_{\boldsymbol{D}} = -\frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N_k}\log P_{\phi}(l_i^{(k)}|\mathbf{h}_i^{(k)})$$

Here l is the domain category and h $\in$ R ^{d} is the hidden representation of both question and passage. The model minimizes Kullback-Leibler (KL) divergence across uniform distribution over K domains. The new loss function uses a new hyper-parameter λ which dicattes the importance of the invariance loss. We've used λ = 0.01 for our experiment.

5.3.3 Final end-to-end model

The training time was approximately 2-3 hours. The time for creating augmented data was approximately 15-20 minutes. Fine-tuning took about 30m-1hr. We trained the baseline using an AdamW optimizer [8]. For the domain adversarial part, we adapted train.py to add finetune with checkpoint model, training for adversarial task and labels. We extended the Distilbert to support DomainDiscriminator. DomainDiscriminator module is taken from Lee et. al [1]. Details of results obtained are given in the subsequent sections.

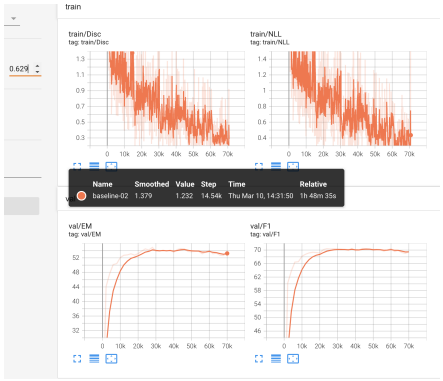
5.4 Results

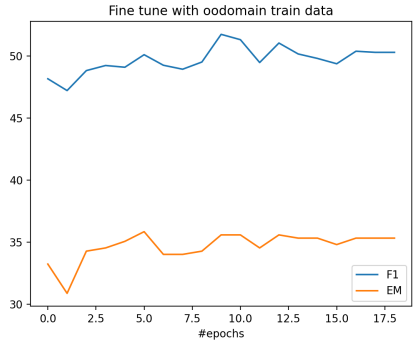
For our experiments, the biggest gain was seen from fine-tuning with oodomain datasets for longer epochs. Our final results are shown in the table below:

	F1	EM
Oodomain Val Set	51.759	35.602
Oodomain Test Set	57.877	40.092

5.4.1 Domain Adversarial Training

Our model for adversarial task trained well over multiple epochs but training time approximately doubled per epoch with later epochs taking longer. As shown below, the loss for both QA task and domain classification task reduced as expected.





Our fine-tuning step focused on the baseline trained model fine-tuned with oodomain train set over multiple epochs and measured the dev set performance for multiple cases. We ran multiple epochs with evaluation after every epoch, multiple epochs with no evaluation but storing the model after every epoch and running few epochs followed by more. The best performance we achieved was for 10 epoch run with no evaluation steps.

5.4.2 Data Augmentation via Back Translation

For the data enhancement, we did the above fine-tuning step with enhanced data with French and Spanish as the pivot languages after trying out few other languages. We got comparable results from the finetuning steps with non-augmented dataset performing better on later epochs as shown in the table below.

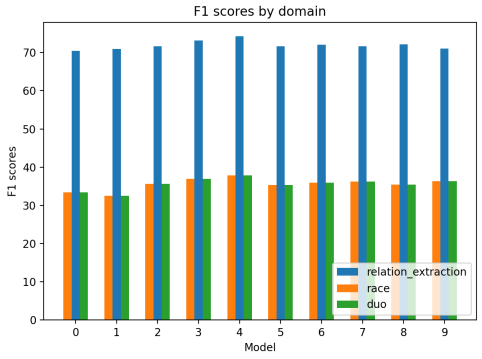
5.4.3 Hyper-parameter Finetuning

We also tried learning rate parameter values of 2.2e-5, 2.5e-5, 3.2e-5 in addition to the default 3e-5. Since the default value gave us the best results, we used this for all of our training runs.

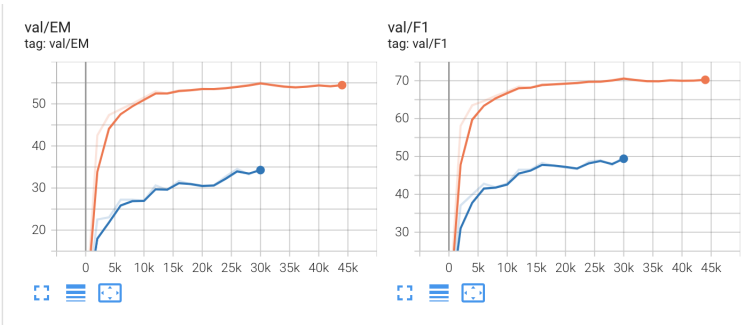
Epochs	1	2	3	4	5	6	7	8	9	10
AugF1	48.77	49.12	49.46	47.41	49	49.39	48.46	49.04	49.39	48.1
BaseF1	48.18	47.23	48.84	49.25	49.11	50.12	49.27	48.95	49.53	51.76

6 Analysis

For the finetuning step with oodomain train set, we also analyzed the performance of various models over different datasets. In general as shown in the figure below, models performed better or worse on all three out of domain datasets i.e., some models generally performed better on all the datasets. Based on this, we focused our optimizing the model for all the three oodomain datasets.



We also observed that the oodomain validation performance was generally better if the indomain validation performance was better for the same model. As such, it was right to choose the right model based on the performance in indomain validation set as shown in the figure below.

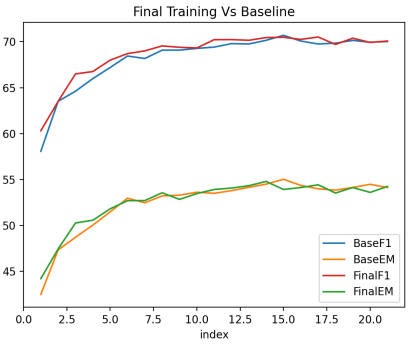


The other significant observation was that we got our best performing models from indomain dataset training from iterations in the earlier epoch. The deeper we went into the epochs, the model performed worse on the oodomain and indomain validation dataset. This confirmed that longer training time does lead to overfit on the train set as seen from the loss reducing over 5 epochs but reduces the performance on generalized data. The models learn the general features towards the beginning of the training epochs.

Also, we saw the model perform better faster on the validation dataset for the adversarial training as compared to the baseline. This can be seen in the figure below. This was perhaps due to two runs through the model in the same iteration to calculate the losses for the discriminator and the QA output. The performance was similar as the training progressed.

7 Conclusion

Finetuning with oodomain dataset gave a significant boost to the performance underscoring the fact that even small domain dataset is pretty significant contributor in model performance.



To obtain a model which is robust to domain shifts, it is important for the model to learn general features and assigning a model additional task may be one way to achieve that. We only tried domain classification as that was straightforward. We also explored sentiment classification but all of the sampled text from the context yielded neutral sentiment using multiple off-the-shelf classifiers. This is one are which we would like to further explore where the text could have some independent property that could generalize across domain and we could classify the data and this would have overlapping features across domains.

We also would like to explore some sophisticated ways to to back translation especially for long contexts with multiple questions and answers that span almost all of the contexts. This could lead to a more robust data enhancement technique. Given the compressed timeframe, we believe that the performance isn't absolutely optimal. However we maintain that these strategies are highly effective in building a Robust QA system and in future we look at further enhancing it to improve the performance of our model.

References

[1] Seanie Lee, Donggyu Kim, and Jangwon Park. Domain-agnostic question-answering with adversarial training. *arXiv preprint arXiv:1910.09342*, 2019.

[2] Shayne Longpre, Yi Lu, Zhucheng Tu, and Chris DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. *arXiv preprint arXiv:1912.02145*, 2019.

[3] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[4] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[5] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[6] Stanford cs 224n robustqa handout. In *http://web.stanford.edu/class/cs224n/ project/default-final-project-handout-robustqa-track.pdf*, 2022.

[7] Backtranslation 0.3.1. In *https://pypi.org/project/BackTranslation/*.

[8] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.