

Improving QA Robustness through Modified Adversarial Training

Stanford CS224N Default Project (Robust)

Jeffrey Propp, Nebyou Zewde

Department of Computer Science

Stanford University

jpropp@stanford.edu, nebyouz@stanford.edu

Abstract

We improved the domain generalizability of a finetuned DistilBert Question Answering (QA) model via adversarial training. By putting a conventional QA model in competition with a discriminator, we generate domain invariant features that will improve the QA model's robustness. We augmented this strategy by pretraining our model on all of our available datasets to gain best performance.

1 Introduction

Though deep learning trained on specific domains has proven very successful in Question Answering (QA) tasks, for real world application it is necessary to build one model that is capable of generalizing to out of domain data. The most accurate existing models, however, do not generalize well and require finetuning or additional training on other datasets. Other methods to improve robustness include data augmentation [1] and task-adaptive finetuning[2]. These methods, however, lack a generalized capability that works with many tasks and without further knowledge of datasets. For these reasons, adversarial training is a promising technique to learn domain invariant features through machine learning.

Research by Lee et al. adapts a technique pioneered by Goodfellow et al. known as adversarial training.[3] Goodfellow et al. used this technique to estimate generative networks in image generation. Lee et al. used the idea of an adversarial framework, and applied it to QA models. Their research aims to build a domain-agnostic QA model that learns domain invariant features through an adversarial training framework. This system consists of a discriminator, which predicts the domain label of a hidden representation, and a QA model, which attempts to make the hidden state indiscernible to the discriminator. We adapted the code used by Lee et al. to modify a baseline Distilbert language model, meshing the existing techniques and experimenting with novel parameters. We trained and validated this method on 6 datasets, using this process to refine the QA model to become domain-agnostic.

2 Related Work

2.1 Pre-trained Language Model

Recently, there have been several applications for using pretrained language models, such as BERT to transfer the knowledge from pre-training to various downstream NLP tasks. BERT [4] is pretrained with bidirectional encoder on large corpora. It randomly masks some input tokens and predicts the masked tokens based on its context. The masked language model enables bidirectional representation, which leads to significant improvements on a number of NLP tasks, such as sentence classification, POS tagging or question answering.

2.2 Domain Generalization

Domain generalization make models more robust to out-of-domain data. The difference between domain adaptation and domain generalization is that for domain generalization, data from the target domain is not available during training

Several methods for domain generalization exist. One of them is to train a model for each indomain dataset. Other approaches break down parameters of a model into domain-specific and domain-agnostic components during training with in-domain dataset, and use the domain invariant parameters for predicting data from unseen target domain. Recently, meta-learning has been proposed for domain generalization.

2.3 Adversarial Training

The idea of adversarial training is originally proposed in the field of image generation [3], known as Generative Adversarial Network (GAN). Adversarial training has since been used for domain adaptation or domain generalization as well.

3 Approach

In our approach to improving QA robustness, we implemented the following:

- We employed adversarial training - a process that attempts to fool AI models by putting them in competition with one another - adapting the method implemented in Lee et al's paper: *Domain-agnostic Question-Answering with Adversarial Training*. [5] We used their code [6] as a framework by which to build our own.
- Our baseline model fine-tunes DistilBERT, a pretrained transformer. The loss function is the sum of the cross-entropy loss for the start and end locations, and we use an AdamW optimizer. We will be testing our model in comparison to how this system performs on out-of-domain datasets.
- Building on this, we added an Adversarial Trainer class, which includes both a QA model and a discriminator model. The discriminator (D) model is trained to classify a joint embedding of question and passage as one of the K in-domain datasets. It does so by minimizing the following cross-entropy loss function, where l is a domain class and $h \in \mathbb{R}^d$ is the combined question and passage hidden representation.

$$L_D = -\frac{1}{N} \sum_{k=1}^K \sum_{i=1}^{N_k} \log P_{\theta}(l_i^{(k)} | h_i^{(k)}) \quad (1)$$

- The QA model attempts to maximize $P_{\theta}(l_i^{(k)} | h_i^{(k)})$, or rather minimize the difference between a normal distribution over K classes (denoted $U(l)$) and the prediction f the discriminator, as shown in the following equation.

$$L_{adv} = -\frac{1}{N} \sum_{k=1}^K \sum_{i=1}^{N_k} (KL(U(l) || P_{\theta}(l_i^{(k)} | h_i^{(k)}))) \quad (2)$$

- The total loss for the QA model is calculated as $L_{QA} + \lambda L_{adv}$, where λ is a hyperparameter that controls the effect of the adversarial loss. Taking note from other research, we set $\lambda = 1e - 2$.
- Training alternates between optimizing the QA model and the discriminator. Our QA model, as in the baseline, is the "distilbert-base-uncased" configuration.
- We ultimately trained our model on 6 available dataset, as this performance was the best. We settled on a batch size of 32, increasing the number of training examples selected each time from 16.
- Through this method of training the QA model against a discriminator, the model learned domain invariant features in order to fool the discriminator, making it more robust when transferred to out-of-domain datasets.

4 Experiments

4.1 Data

We trained our model using the Stanford Question Answering Dataset (SQuAD) [7], Natural Questions [8] and NewsQA [9], preprocessed in the same format as SQuAD. For some trials, we also trained on parts of RelationExtraction [10], DuoRC [11], and RACE [12] datasets. We evaluate our model on test examples from these three out-of-domain datasets: RelationExtraction, DuoRC, and RACE.

4.2 Evaluation method

We use the Exact Match (EM) score and F1 score metrics to evaluate the performance of our model. Exact Match is a binary measure (i.e. true/false) of whether the system output matches the ground truth answer exactly. F1 is a less strict metric – it is the harmonic mean of precision and recall. When evaluating on the validation or test sets, we take the maximum F1 and EM scores across the three human-provided answers for that question.

4.3 Experimental details

We ran all our tests with the "distilbert-base-uncased" configuration and a learning rate of $3e-5$.

Baseline: Our first iteration of adversarial training (Adv-01) was trained on the three in-domain datasets but run with the incorrect number of classes for the discriminator to guess from. We thus received poor performance. Batch size was 16. Runtime was 16h 18m.

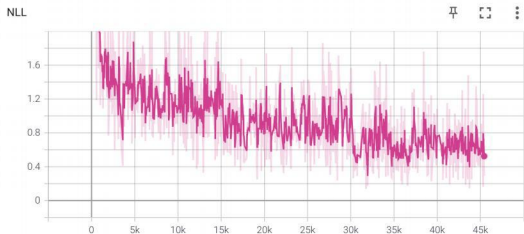


Fig 1: Baseline training NLL (Negative Log-Likelihood) loss by Step

Adversarial Trial 1: Our first iteration of adversarial training (Adv-01) was trained on the three in-domain datasets but featured a bug in loading the saved model from its checkpoint. We thus received poor performance. Batch size was 16. Runtime was 22h 57m.

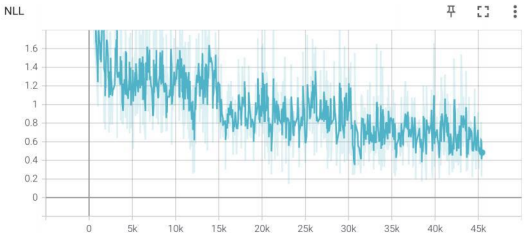


Fig 2: Adv-01 training NLL loss by Step

Adversarial Trial 2: Our second iteration of adversarial training (Adv-02) was trained on three datasets but was run with the incorrect number of classes, as we input 6 rather than the 3 datasets it was trained on. Batch size was 16. These results were slightly below the baseline. Runtime was 21h 47m.

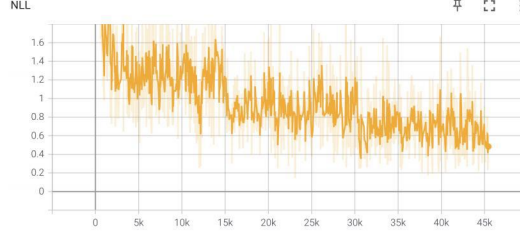


Fig 3: Adv-02 training NLL loss by Step

Adversarial Trial 3: Our third iteration of adversarial training (Adv-03) was trained on six datasets, from in domain and out of domain, and was run with the correct number of classes, as we input 6. Batch size was 16. These results were slightly above the baseline. Runtime was 23h 52m.

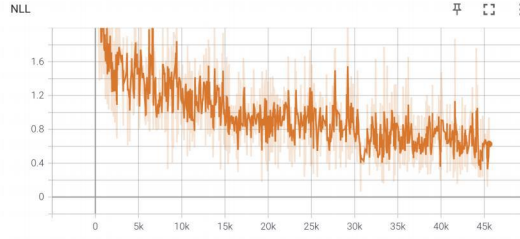


Fig 4: Adv-03 training NLL loss by Step

Adversarial Trial 4: The last iteration of adversarial training (Adv-04) was trained on six datasets, from in domain and out of domain, and was run with the correct number of classes. We also changed batch size to 32, increasing the training examples used per iteration. We altered our labels to randomly generated numbers within the class range for each batch. These changes greatly improved our model's performance, especially on the test set. Runtime was 20h 2m.

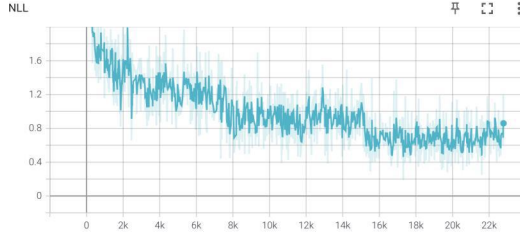


Fig 5: Adv-04 training NLL loss by Step

4.4 Results

	Baseline	Adv-01	Adv-02	Adv-03	Adv-04
EM	33.246	0.000	32.460	34.031	35.864
F1	48.432	4.251	48.050	48.639	50.382

	Baseline	Adv-01	Adv-02	Adv-03	Adv-04
EM	-	-	-	41.422	43.028
F1	-	-	-	59.481	60.071

Fig 6: Model performance on validation and test sets.
Above is validation set (out of domain) and below is test set (out of domain).

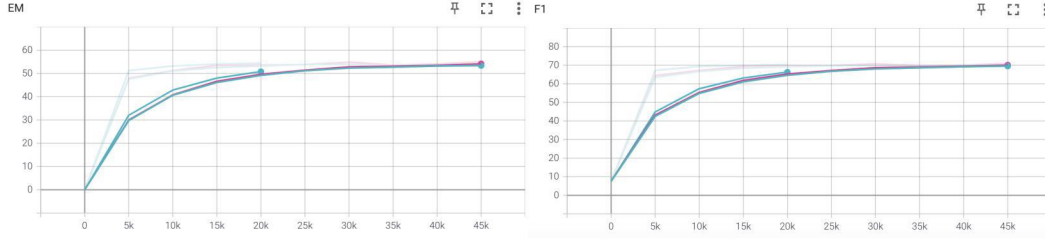


Fig 7: Model EM and F1 scores over time

Our initial results were quite disappointing, but ultimately understandable given the error in loading the saved model and the subsequent error in the number of classes. Having the incorrect number of classes means our discriminator would initially guess the answer was from datasets that the model had not been trained on. Our revised model, with the increased number of training sets, barely improved on the performance of the baseline. We believe this markedly small improvement derives from how we designed the labels to sequentially generated numbers within the class range for each batch.

Our performance improved when we corrected these labels to be randomly generated numbers within the class range for each batch and made the batch size 32. We believe that the labels accounted for the majority of this improvement, as randomly generated numbers more accurately represent a normal distribution, which is the distribution that the discriminator should be tested against for loss, as we ultimately want it to randomly guess the class of the answer with no pattern or information.

5 Analysis

Our model's performs better than the baseline and we have not seen it perform in any unexpected ways.

We seem to obtain a better domain-agnostic QA model that is capable of handling out-of-domain data when labels used to obtain the discriminator loss are randomized in a normal distribution. By using truly randomized labels to calculate loss from the predicted domain label of hidden representation from QA model, our model does a better job at learning domain invariant features.

From our model, we observe that a lot of the incorrect answer predictions contain the correct answer as a subset of the output. Some examples are illustrated below:

- **Question:** where was the first t20 cricket match played
- **Context:** BPB Twenty20 cricket , sometimes written Twenty - 20 , and often abbreviated to T20 , is a short form of cricket . At the professional level , it was originally introduced by the England and Wales Cricket Board (ECB) in 2003 for the inter-county competition in England and Wales . In a Twenty20 game the two teams have a single innings each , which is restricted to a maximum of 20 overs . Together with first - class and List A cricket , Twenty20 is one of the three current forms of cricket recognised by the International Cricket Council (ICC) as being at the highest international or domestic level . A typical Twenty20 game is completed in about three hours , with each innings lasting around 75 – 90 minutes and a 10 – 20 - minute interval . This is much shorter than previously - existing forms of the game , and is closer to the timespan of other popular team sports . It was introduced to create a fast - paced form of the game which would be attractive to spectators at the ground and viewers on television . EEPE
- **Answer:** in 2003 for the inter-county competition in England and Wales
- **Prediction:** England and Wales

Fig 8: adv-04 model text output example

- **Question:** which church was given a letter in the book of revelation
- **Context:** BPB In an early part of the Revelation , on the Greek island of Patmos , Jesus Christ instructs John of Patmos to : `` Write on a scroll what you see and send it to the seven churches : to Ephesus , and to Smyrna , and to Pergamum , and to Thyatira , and to Sardis , and to Philadelphia , and to Laodicea . " EEPE
- **Answer:** Ephesus
- **Prediction:** see and send it to the seven churches : to Ephesus , and to Smyrna , and to Pergamum , and to Thyati

Fig 9: adv-04 model text output example

Some incorrectly predicted answers are a subset of the correct answer phrases. Examples are illustrated below:

- **Question:** when did holland become involved in world war 2
- **Context:** BPB The direct involvement of the Netherlands in World War II began with its invasion by Nazi Germany on 10 May 1940 . The Netherlands had proclaimed neutrality when war broke out in September 1939 , just as it had in World War I , but Adolf Hitler ordered it to be invaded anyway . On 15 May 1940 , one day after the bombing of Rotterdam , the Dutch forces surrendered . The Dutch government and the royal family escaped and went into exile in London . EEPE
- **Answer:** with its invasion by Nazi Germany on 10 May 1940
- **Prediction:** 10 May 1940

Fig 9: adv-04 model text output example

- **Question:** Where are pyrenoids found?
- **Context:** The chloroplasts of some hornworts and algae contain structures called pyrenoids. They are not found in higher plants. Pyrenoids are roughly spherical and highly refractive bodies which are a site of starch accumulation in plants that contain them. They consist of a matrix opaque to electrons, surrounded by two hemispherical starch plates. The starch is accumulated as the pyrenoids mature. In algae with carbon concentrating mechanisms, the enzyme rubisco is found in the pyrenoids. Starch can also accumulate around the pyrenoids when CO₂ is scarce. Pyrenoids can divide to form new pyrenoids, or be produced "de novo".
- **Answer:** The chloroplasts of some hornworts and algae
- **Prediction:** chloroplasts

Fig 10: adv-04 model text output example

We theorize that span refinement could be a concept that can improve our predictions if applied. In most baseline models, only the first occurrence of answer text is used for training for the sake of simplicity. However, if we take the semantic and context of the question into better consideration, a certain phrase in the passage may be more likely to a better answer fit relevant to the question. In order to find the most plausible answer span, a question and sentences in the passage are encoded into fixed-size vectors with universal sentence encoder. We think that such a framework can give our model more flexibility and possibly help identify more suitable answer spans.

6 Conclusion

Through our experimentation in adopting an existing adversarial training method to DistilBERT Question-Answering tasks, we have learned that adversarial learning is a viable method for improving generalization of language models. We were able to gain favorable results on the out of domain test set, highlighting the success of our model in learning domain invariable features by attempting to maximize discriminator loss.

Our research has its limitations. We built only on existing research, and therefore our findings were restricted to methods. We changed multiple features per trial, and so cannot attribute the model's success to one specific change. Finally, we did not experiment heavily with other methods of improvement, so room remains to see what other techniques for improving robustness could be combined with adversarial training.

By analyzing the architecture of our model and understanding the framework of the technique, we know that this technique is transferable to other domains of machine learning, such as gameplay and image generation. To refine our own model, we could further experiment with hyperparameters and loss. In future re work, we could apply this method to other types of models, or experiment with greater numbers of datasets and more distinctive datasets.

References

- [1] Zhucheng Tu Shayne Longpre, Yi Lu and Chris DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. In *AarXiv preprint*, 2019.
- [2] Swabha Swayamdipta Kyle Lo Iz Beltagy Doug Downey Suchin Gururangan, Ana Marasović and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. In *arXiv preprint arXiv:2004.10964*, 2020.
- [3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, 2014.

- [4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.
- [5] Jangwon Park Seanie Lee, Donggyu Kim. Domain-agnostic question-answering with adversarial training. In *Proceedings of the Second Workshop on Machine Reading for Question Answering*, 2019.
- [6] <https://github.com/seanie12/mrqa>.
- [7] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016.
- [8] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.
- [9] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, August 2017. Association for Computational Linguistics.
- [10] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension, 2017.
- [11] Amrita Saha, Rahul Aralikkatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. Duorc: Towards complex language understanding with paraphrased reading comprehension, 2018.
- [12] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017.