# Implementing Domain Adversarial Training with Data Augmentation for Out-of-Domain Robust Question Answering

Stanford CS224N Default Project
Track: RobustQA

**Thomas Le Menestrel**
ICME Department
Stanford University
tlmenest@stanford.edu

**German Kolosov**
MS&E Department
Stanford University
gkolosov@stanford.edu

**Juan Langlois**
MS&E Department
Stanford University
jlangloi@stanford.edu

## Abstract

The goal is to build a robust question answering (QA) system by fine-tuning the pre-trained Transformer encoder architecture DistilBert [1]. By a robust QA system, we mean that the system can perform well in out-of-domain datasets. In particular, the goal is to do better than a baseline. To accomplish this, we propose using adversarial training to fine-tune the encoder. The model training will alternate between two alternative objectives, correctly answering the question and classifying the domain. After the training, the model will be fine-tuned using a small out-of-domain sample.
After submitting the **Adversarial + Fewshot** model to the test leaderboard, we obtained the following results: **EM: 40.321, F1: 58.229, Validation Rank 16/62, Test Rank 37/50**.

## 1 Key Information to include

- TA mentor: Kaili Huang
- External collaborators, External mentor, Sharing project : No
- For more details see the code

## 2 Introduction

Critical to Deep Learning's success in real-world applications is the ability to build models that generalize well to unseen data and can therefore cover multiple domains. In particular, in NLP Question Answering (QA) tasks, models struggle to generalize and often over-fit a specific dataset, making them unreliable for tasks on new, poorly labeled datasets. It is therefore crucial to develop methodologies to train domain-agnostic QA models. In other words, these models should be able to learn domain-invariant features that can generalize to unseen data.
In this project, we will explore building such a QA system that can adapt to unseen domains with only a few training samples from the domain. We propose using an Adversarial Training framework, similarly to GANs in computer vision [2]. Finally, to further increase the robustness, we use Easy Data Augmentation [3] to augment our small sample of out-of-domain training data.

## 3 Related Work

An Adversarial Training framework's goal is to learn domain invariant features that do not encode spurious information. Concretely, given an input $x$ from a domain $d_i$, the features extracted by the

model $f(x)$ are fed into a discriminator $g$ that attempts to classify the domain of $x$. This technique trains $g$ to maximize the probability of predicting the correct domain. Model $f$, on the other hand, requires a signal to maximally "confuse" the discriminator.

Introducing adversarial training to improve NLP tasks has been explored in tasks other than QA, including sentiment analysis [4, 5] as well as relation extraction [6]. For QA, [7] apply this to the BERT-based QA setting by training $g$ to identify the representations output by BERT at the `[CLS]` token.

# 4 Approach

To achieve a QA system that will be robust to the out-of-domain questions, we will finetune an adversarial model [7] on top of a pre-trained transformer [8]. The role of the adversarial model is to learn domain agnostic representations by penalizing domain-specific representations. We also plan to use data augmentations [9] to improve learning when data is scarce.

We can define the problem as follows:
*Given the $K$ in-domain datasets $\mathcal{D}_i$, consisting of triplets of passage $\mathbf{c}$, question $\mathbf{q}$, and answer $\mathbf{y}$, where $\mathcal{D}_i = \{\mathbf{c}_i^{(k)}, \mathbf{q}_i^{(k)}, \mathbf{y}_i^{(k)}\}_{i=1}^{N_k}$. The model learned from $\{\mathcal{D}_i\}_{i=1}^{K}$ predicts $\mathbf{y}_j^l$ from $\mathbf{c}_j^l$, $\mathbf{q}_j^l$ for each $L$ out-of-domain datasets $\{\mathcal{D}_j\}_{j=1}^{L}$.*

## 4.1 Baseline

The baseline model consisted of the pre-trained DistilBert [1] model adapted for a Question Answering task. In particular, we used the Huggingface implementation `DistilBertForQuestionAnswering`, which is a DistilBert Model with a span classification head. The pre-trained encoder functions as a feature extractor, while the span classification head is for extractive question-answering tasks. More specifically, a linear layer on top of the hidden-states output of the DistilBert model will compute span start and end logits. Finally, we use cross-entropy loss to compare the predicted distributions of the span's end and start location with a target one-hot discrete distribution:

$$\mathcal{L}_{QA} = -\frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N_k}\left[\log P_\theta(\mathbf{y}_{i,s}^{(k)} \mid \mathbf{x}_i^{(k)}, \mathbf{q}_i^{(k)}) + \log P_\theta(\mathbf{y}_{i,e}^{(k)} \mid \mathbf{x}_i^{(k)}, \mathbf{q}_i^{(k)})\right] \tag{1}$$

Where $N$ is the total number of samples, $\mathbf{y}_{i,s}$ is the starting position of passage and $\mathbf{y}_{i,e}$ is the end position of answer in the passage. The training process optimizes the linear head as well as fine-tunes the DistilBert model.

## 4.2 Domain Adversarial Training

The Adversarial model is based on the architecture proposed by [7] (Figure 1), which uses a Domain Adversarial Training framework inspired by the GAN model [2]. Specifically, we will jointly train a span classification head and a discriminator head. We can train both heads by using hidden states from the last layer of a pre-trained DistilBert model. The idea is to progressively mislead the discriminator by learning domain agnostic features without hurting the QA performance. The training proceeds by iteratively training the transformer and QA head on an augmented QA loss and the discriminator head on a simple cross-entropy loss. Given the discriminator head $D$ and the problem definition, the latter loss is:

$$\mathcal{L}_D = -\frac{1}{N}\sum_{k=1}^{K}\sum_{i=1}^{N_k}\log P_\phi(l_i^{(k)} \mid \mathbf{h}_i^{(k)}) \tag{2}$$

Where $l$ is the domain category and $\mathbf{h} \in \mathbb{R}^d$ is the hidden representation of both question and passage. In this case, we use DistilBert's last layer hidden representation for the token `[CLS]` as $\mathbf{h}$.

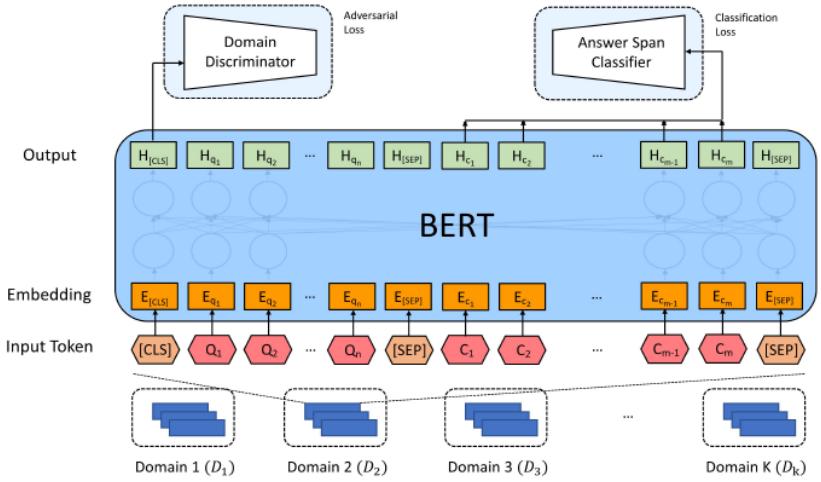



Figure 1: Overall training procedure for learning domain-invariant feature representation. Model learns to predict start and end position in the passage and fool discriminator for domain-invariant representation. Source figure: [7].

The augmented QA loss is going to try to minimize the loss in equation 1 and maximize the cross-entropy loss from 2. The latter is equivalent to minimizing the Kullback-Leibler (KL) divergence between uniform distribution over $K$ classes denoted as $\mathcal{U}(l)$ and the discriminator's prediction:

$$\mathcal{L}_{adv} = \frac{1}{N} \sum_{k]1}^{K} \sum_{i=1}^{N_k} KL(\mathcal{U}(l) \| P_\phi(l_i^{(k)} \mid \mathbf{h}_i^{(k)})) \tag{3}$$

The final loss for the QA model is $\mathcal{L}_{QA} + \lambda \mathcal{L}_{adv}$ where $\lambda$ is a hyper-parameter for controlling the importance of the adversarial loss.

$$L_{global} = L_{QA} + \lambda L_{adv} \tag{4}$$

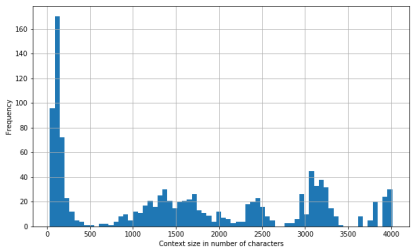
As a starting point for the implementation, we studied the code provided by the authors of [7] and stuck with their 3-layer perceptron architecture for the discriminator.

### 4.3 Easy Data Augmentation

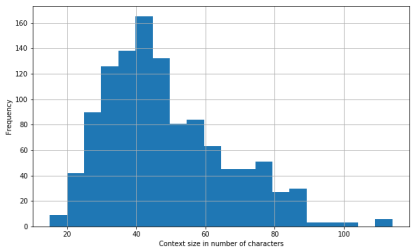
Once our Adversarial QA DistilBert model is trained on the main training set, it is supposed to have learned some general features that should be transferable to other domains. Although the model can be used directly for a task of out-of-domain QA, finetuning it on a sample of out-of-domain sets, if available, would increase the performance of the QA system on that specific dataset.
The challenge with these datasets would often be the small size of labeled data, which is our case 1. To overcome this obstacle, we proceeded to augment the out-of-context training data.

**Data Augmentation in NLP :** First, we need to chose the type of the augmentations that should be used. There exists an array of data augmentation techniques of varying complexities in NLP that are commonly used to increase the performance of language models.
A lot of these techniques are task-agnostic and can be applied easily to our QA datatset. Some use external knowledge like *synonym replacement* for example, others like *random swap* and *random deletion* don't require any external information.

In our case, we chose to implement some of the methods explained in 'Easy Data Augmentation' [3] which include **Synonym replacement** and **Random Deletion**.




(a) Distribution of the context sizes




(b) Distribution of the question sizes

Figure 2

**Which part to augment ?** Given our task, we can either augment the context, the question or both of them. Given the asymmetric nature of the context's and the questions' sizes, as seen in figures 2a and 2b, it made more sense for us to only augment the context. As introduced before, the following augmentations were implemented and applied to the context texts:

1. **Synonym replacement** Every word is replaced by a randomly chosen synonym with a probability $p_{SR}$.
2. **Random Deletion** Every word is deleted with probability $p_{RD}$

Both of the augmenting techniques mentioned above, are applied only to the part of the context that does not contain the answer. The data is augmented $N$ times, meaning that if the dataset contains M samples, we end up with $NxM$ samples.

Let's see an example of the EDA, with the parameters $p_{SR}, p_{RD}, N = 0.5, 0.2, 2$ on 'duorc' dataset [10].

- **Original Context:** *"The film depicts the construction and ultimate demolition of a metaphorical wall. "*
- **Augmented Context 1:** *"The movie describe the construction and of a rampart. "*
- **Augmented Context 2:** *"depicts construction and ultimate a wall "*

We can see that the the word 'film' was swapped for 'movie', 'wall' for 'rampart', 'depicts' for 'describe', and the words 'ultimate', 'demolition' have been deleted. The meaning of the sentence might have slightly changed but we still understand the overall context.

The parameters $p_{SR}, p_{RD}, N$ are crucial to the task and should be chose carefully. A value that is too low would change close to nothing in our dataset and therefore the performance's model but high values would drastically transform and/or delete the quasi-entirety leading to a decline in performance, which we witnessed during our training.

# 5 Experiments

## 5.1 Data

For this project we use three *in-domain* reading comprehension datasets (SQuAD [11], NewsQA [12] and Natural Questions [13]) and three *out-of-domain* datasets (Relation Extraction [14], DuoRC [10] and RACE [15]). We split each in-domain dataset into two sets: training and validation. On the other hand, we split the out-of-domain datasets in three: training, validation, and test. We use the in-domain datasets for the bulk of the training while we use the out-of-domain datasets for few-shot training (also what we called finetuning), evaluation, and testing. The statistics for each dataset are summarized in table 1.

| Dataset | Question Source | Passage Source | Train | dev | Test |
|---|---|---|---|---|---|
| In-Domain Datasets | | | | | |
| SQuAD [11] | Crowdsourced | Wikipedia | 50000 | 10507 | - |
| NewsQA [12] | Crowdsourced | News articles | 50000 | 4212 | - |
| Natural Questions [13] | Crowdsourced | Wikipedia | 50000 | 12836 | - |
| Out-of-Domain Datasets | | | | | |
| Relation Extraction [14] | Crowdsourced | Movie reviews | 127 | 126 | 1248 |
| DuoRC [10] | Teachers | Examinations | 127 | 128 | 419 |
| RACE [15] | Synthetic | Wikipedia | 127 | 128 | 2693 |

Table 1: Statistics for datasets used for building the QA system for this project. Question Source and Passage Source refer to data sources from which the questions and passages were obtained.

To better clarify the types of datasets we are dealing with as well as the exact nature of the QA task, let's take an example from the SQuAD [11] dataset. It contains a context, a question and the span of the answer which is the ground truth :

- *Context :* Warsaw (Polish: Warszawa) is the capital and largest city of Poland. It stands on the Vistula River in east-central Poland, roughly 260 kilometres (160 mi) from the Baltic Sea and 300 kilometres (190 mi) from the Carpathian Mountains.
- *Question :* What is the largest city of Poland?
- *Answer Span : Warsaw*

### 5.2 Evaluation method

We will evaluate the models in the held out test set of the out-of-domain datasets. The performance of each model is measured by two metrics:

- Exact Match (EM) : a binary measure that shows if the output of the system matches the true answer. Its binary nature makes it a very strict metric.
- F1 score : the harmonic mean of precision and recall. In the case of QA, F1 score measures the overlap of the answer provided compared to the ground truth, which makes it a looser metric compared to the EM.

For the evaluation, we take the maximum F1 and EM scores across the three human-provided answers for that question averaged over the entire dataset.

### 5.3 Experimental details

We ran experiments at two levels or stages:

- (i) in-domain train and validation
- (ii) few-shot out-of-domain train and validation

The first level experiments help us measure the generalization performance of each model. On the other hand, the second-level experiments help us evaluate the effects of few-shot training on each model. First, we finetuned the two models on the in-domain training set and saved the models that performed best on the in-domain validation set. The second-level models result from further training the first-level models on the out-of-domain set and saving the models that performed best on the out-of-domain validation set.
In terms of hyperparameters, we start from HuggingFace's "distilbert-base-uncased" DistilBert pre-trained model and train with a learning rate of $3e^{-5}$ and a batch size of 32. Additionally, for the Adversarial models, we required the hyperparameter $\lambda$, which we set at $1e^{-2}$ as recommended by [7]. Another important hyperparameter was the evaluation frequency for model selection. The few-shot training is done with a smaller learning rate of $3e^{-6}$ and over 1 epoch. For Data Augmentation, we use the following parameters : $p_{SR}, p_{RD}, N = 0.3, 0.1, 2$
As mentioned before, during training, we selected the models that performed best on the corresponding validation set. While doing in-domain training, we evaluated each model on the validation set every

5000 steps. On the other hand, since the out-of-domain data was scarce, we evaluated each model every 10 steps.

**In-domain train and validation models:**

- **Baseline:** pre-trained DistilBert model with a Question Answering head fine-tuned to the in-domain data.
- **Adversarial:** pre-trained DistilBert model with the adversarial architecture fine-tuned to the in-domain data.

**Few-shot out-of-domain train and validation models:**

- **Baseline + Fewshot:** Baseline model trained on small train set of out-of-domain data.
- **Adversarial + Fewshot + Freeze DistilBert:** Adversarial model trained on small train set of out-of-domain data while freezing the DistilBert parameters.
- **Adversarial + Fewshot:** Adversarial model trained on small train set of out-of-domain data.
- **Adversarial + Fewshot + EDA:** Adversarial model trained on small train set of augmented out-of-domain data.

### 5.4 Results

Figures 3 and 4 show the loss trajectories for the models trained on in-domain and out-of-domain data, respectively. These plots yield two interesting insights.
First, in the left subfigure of Figure 3, we can see that the Adversarial model's Discriminator loss has a big initial improvement, but then it quickly starts increasing. This was the case even when we used a small $\lambda$ value such as $1e-2$.

Furthermore, if we look at the trajectories in Figure 4, we can see from the blue curve that Adversarial model cannot learn in the few-shot exercise without the help of the transformer. This observation supports the claim that most of the model's power comes from the transformer rather than any actual Adversarial training.

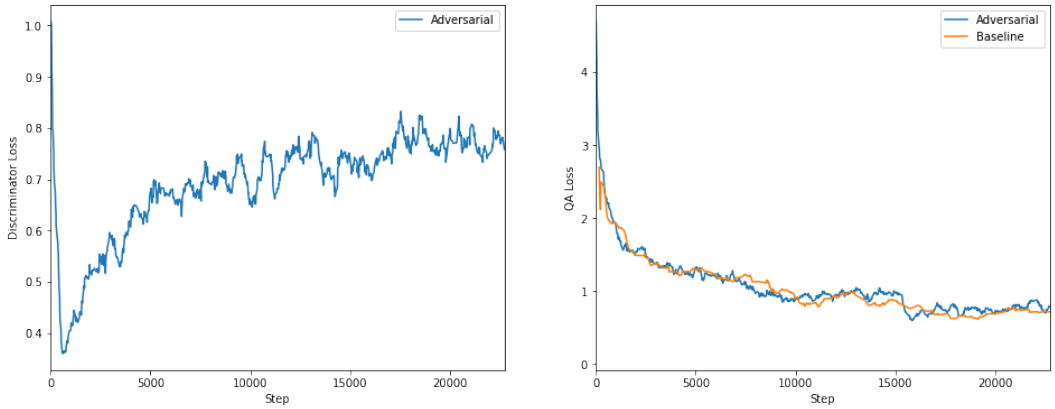



Figure 3: Smoothed loss of in-domain training.

Table 2 summarizes the results obtained by each model on the out-of-domain validation set. From the first stage of training, we can see that the Adversarial and Baseline models have similar performance. However, using few-shot training obtains significant performance improvements. In particular, from the results, we can see that few-shot training only works when we continue to train the pretrained transformer parameters. Easy Data Augmentation seems to help performance but only slightly. Finally, we submitted the best performing model, i.e., Adversarial + Fewshot + EDA, to the test leaderboard and obtained the following results: **EM: 40.321, F1: 58.229, Validation Rank 16/61, Test Rank 37/50**.

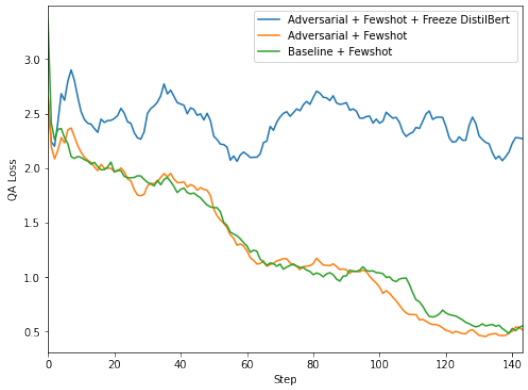



Figure 4: Smoothed loss of few-shot out-of-domain training.

| Model | EM | F1 |
|---|---|---|
| Baseline | 32.46 | 49.31 |
| Adversarial | 31.94 | 48.48 |
| Adversarial + Fewshot + Freeze DistilBert | 31.94 | 48.48 |
| Adversarial + Fewshot | 35.864 | 50.523 |
| Adversarial + Fewshot + EDA | **37.173** | **51.383** |

Table 2: Results summary for baseline and adversarial RobustQA models

## 6 Analysis

As highlighted in the previous section, the adversarial component of our model didn't exactly improve its performance. The observation of the increasing loss after it decreased at the beginning, summed with the fact that the QA loss of the Adversarial model is similar to the QA loss of the Baseline model, suggests that the 3-layer perceptron was not a powerful enough model for the Adversarial Training task. In retrospect, this is understandable because the QA head has the whole transformer helping minimize the adversarial loss, clearly overpowering the discriminator.
Concerning the comparison between the different models, it seems that the Adversarial model and the Baseline Model have learned similar features, which means that the Adversarial model did not acquire domain-agnostic knowledge as hypothesized. This is explained, as mentioned before, by the fact that the QA head alone is too weak of a model to adjust to the new data. Few-Shot training coupled with Data Augmentation was the best performing model so far, which is understandable because, during this fine-tuning the model acquires some domain-specific features that helps in the task. Moreover, for few-shot training, the use of a smaller learning rate helps carefully adjust the weights already learned by the QA model to the unseen dataset, which is why the performance is much better.
Concerning the drop between Validation and Test Scores that is clearly reflected in our respective ranks, we think that it might be stemming from overfitting our parameters on the validation set which seems to be a bit different from the test set.

## 7 Conclusion

Through this project, we explored the usefulness of an adversial setting in QA as well as the role of few-shot training using out-of-domain data. Concerning the former, although our results weren't satisfying, we think that the idea still has potential in improving out-of-domain predictions. A suggestion would be to use a more powerful model as our discrimator, through adding an encoder or more layers and neurons.
Moreover, we have showed that augmenting the scarce out of domain data with simple augmentation techniques increases the robustness to the unseen domain.
Finally, the few-shot training on the out-of-domain datasets, for less epochs and with a smaller dataset, makes the model adapt to the new domain and increases the performance with only a few iterations.

## References


[1] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. 2020.

[2] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. 2014.

[3] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. 2019.

[4] Xilun Chen, Ben Athiwaratkun, Yu Sun, Kilian Q. Weinberger, and Claire Cardie. Adversarial deep averaging networks for cross-lingual sentiment classification. *CoRR*, abs/1606.01614., 2016.

[5] Xilun Chen and Claire Cardie. Multinomial adversarial networks for multi-domain text classification. 2018.

[6] Yi Wu, David Bamman, and Stuart Russell. Adversarial training for relation extraction. 2017.

[7] Seanie Lee, Donggyu Kim, and Jangwon Park. Domain-agnostic question-answering with adversarial training. *CoRR*, abs/1910.09342, 2019.

[8] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi. Revisiting few-sample bert fine-tuning, 2021.

[9] 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.

[10] Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. Duorc: Towards complex language understanding with paraphrased reading comprehension. *ACL*, 2018.

[11] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *abs/1606.05250*, 2016.

[12] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bach-man, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *ACL 2017, page 191*, 2017.

[13] 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. *Association for Computational Linguistics (ACL)*, 2019.

[14] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. *arXiv:1706.04115*, 2017.

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