Improving the Robustness of QA Systems through Data Augmentation and Mixture of Experts

Stanford CS224N {Default} Project

Ruying Gao
Department of Management Science and Engineering
Stanford University
rgao1@stanford.edu

Mei Hao
Department of Mechanical Engineering
Stanford University
mhao@stanford.edu

Zhengqiu Lou
Department of Mechanical Engineering
Stanford University
lou1996@stanford.edu

Mentor: Elissa Li | External Collaborators: N/A | Sharing Project: No

Abstract

Despite the stunning achievements of question answering (QA) systems in recent years, existing neural models tend to fail when they generalize beyond the in-domain distributions. This project seeks to improve the robustness of these QA systems to unseen domains through a combination of Easy Data Augmentation (EDA) and Mixture of Experts (MoE) techniques. As baseline, we finetuned a pre-trained DistilBERT model with Natural Questions, NewsQA and SQuAD datasets using the default configurations and evaluated the model performance on the out-of-domain datasets, including RelationExtraction, DuoRC, and RACE. After obtaining our second baseline by including a small number of training examples from our out-of-domain datasets, we ran two rounds of hyperparameters tuning through random search. Based on the best performing set of hyperparameters, we then augmented our out-of-domain datasets using the EDA techniques and analyzed the effects of each technique through a series of experiments. Finally, we implemented an MoE model with three experts and a two-layer bi-directional LSTM followed by a linear layer as the gating function. Both the data augmentation technique and the mixture-of-expert approach demonstrated capability to improve the robustness of DistilBERT-based QA systems, and a combination of the two methods brings even further improvement. The combined approach increased the F1 and EM scores on the dev set by 15.03% and 14.87%, respectively, compared to the baseline, and achieved an F1 score of 62.062 and an EM score of 42.317 on the test leaderboard.

1 Introduction

Since the birth of the Stanford Question Answering Dataset (SQuAD) in 2016, huge progress has been made in the field of natural language processing (NLP) for reading comprehension and question answering. From the earlier DCN, BiDAF, and LSTM-based architectures, to more recent Transformer-based models[1][2][3][4], large neural network models are achieving better performance on standard question answering tasks, becoming on par and even superior to human performance.

Despite these stunning accomplishments, existing neural network models do not yet seem to have the capability to acquire transferable reading skills [5]. Unlike humans, who can easily complete similar tasks in different settings, neural network models are vulnerable to adversarial inputs [6] and tend to fail when asked to generalize beyond the in-domain distributions [7]. Such fragility poses a daunting challenge to the deployment of NLP systems in the real world, where the training data and test data often do not come from the same distributions.

Various techniques and novel architectures have been developed by researchers around the world to tackle this problem. In this project, we seek to make our contributions by combining techniques from advances in other NLP areas and building a QA system that is robust to out-of-domain tasks. Specifically, our team adapted the Easy Data Augmentation (EDA) techniques from Wei and Zou (2019)[8], implemented a mixture-of-experts model with three experts and a two-layer bi-directional LSTM followed by a linear layer as the gating function, and analyzed the results for using a combination of these techniques. Both approaches demonstrated some potential for improving the system performance on out-of-domain datasets, with a combination of the two most effective.

2 Related Work

Researchers have explored the efficacy of including more adversarial inputs in the training corpus to improve the robustness of NLP systems [9][10]. One such method is to augment the training samples with meaning-preserving perturbations, which has been shown to successfully ameliorate the brittleness of neural network models. Registering invariant features of the samples, the model becomes less likely to rely on superficial correlations. The approach that our team took was adapted from Wei and Zou (2019) [8], which introduces the easy data augmentation (EDA) techniques that modify data at a word level. This method is relatively easy to implement, but it falls short in producing semantically equivalent paragraphs and sometimes produces sentences that are grammatically incorrect or meaningless to humans. Works described in Ribeiro *et al.* (2018) [9] improves upon that by introducing semantically equivalent adversarial rules for the samples. One other prevalent method for data augmentation relies on back translation [10], which involves translating samples in the current language to a pivot language, and back-translating the samples to obtain paraphrases of the original samples.

Another class of methods that focuses on the model itself is the mixture-of-expert (MoE) approach. This approach was first described in detail by Jordan and Jacobs (1991)[11]. The initial model contains a simple tree-like architecture that involves three major components: 1) multiple experts, 2) a gating function that partitions the input space and decides which expert output is reliable in different regions, and 3) a probabilistic model that combines the gating values and the expert outputs [12]. Such an architecture allows differentiated assignments: different inputs could be assigned to different "experts" and be evaluated with the most appropriate methods. We found this approach to be particularly promising for our task: by using a gating function to carefully control the weights given to each expert for a sample, the experts can hopefully learn different distributions of the out-of-domain datasets and give better predictions. Seeing that we only have a small number of out-of-domain training examples, we decided to replicate the EDA approach and study if the variations introduced by the augmented data can give the model new information and allow it to rely less on superficial correlations, thus further improving the MoE model performance.

3 Approach

3.1 Baseline

Our basic baseline was established by a pretrained DistilBERT model [13] finetuned with data from our in-domain datasets, namely, Natural Questions, NewsQA and SQuAD.

3.2 Data Augmentation

Our first attempt to improve the stability of our model is to incorporate the out-of-domain (OOD) data edited by simple techniques into the finetuning process. Specifically, we augmented the OOD datasets with random synonym replacement (SR), random deletion (RD), random insertion (RI), and random swap (RS) techniques based on the Easy Data Augmentation (EDA) methods from Wei and Zou (2019)[8]. Our team adapted the code snippets that perform specific SR, RD, RI and RS tasks from [8] to fit our task of question answering (`https://github.com/jasonwei20/eda_nlp`). In particular, to ensure that we have the right index for the answers after editing the context, we wrote three control functions. The first extracts the context from each sample. The second divides the sample context into three parts: before answer (context 1), answer (context 2) and after answer (context 3). Afterwards, this function calls a pre-specified data augmentation function on context 1 and 3 with a user-defined target percentage of adjusted context. The third function calculates the new starting position of the answer based on the length of context 1 and combines the divided contexts to form a new augmented sample. This process is shown in Figure 1.

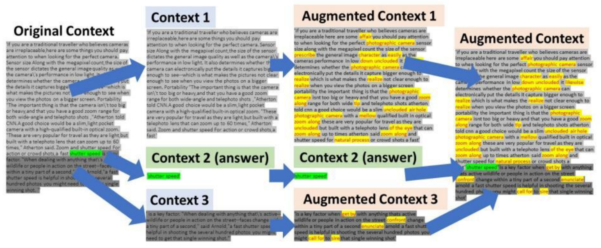


Figure 1: Context Division for Data Augmentation

3.3 Mixture of Experts

The mixture-of-experts method was implemented through two approaches: first, we tried out a naive model without a gating function. Specifically, instead of training a single model on three out-of-domain datasets, we trained three separate DistilBERT models simultaneously, each on a different out-of-domain dataset (the starting point of all three models is our baseline model). During the evaluation process, examples from the three out-of-domain datasets are fed into their corresponding models to generate predictions. For performance evaluation, predictions from the three models are combined to calculate the EM and F1 scores.

Next, we tried out the more complex approach in alignment with the design from Jacobs *et al.* [11]. We again trained three DistilBERT model based on the baseline model as our three experts. However, instead of manually feeding examples into their corresponding models, we designed a gating function which processes the inputs and decides how much weight to give each expert in formulating predictions, as shown by Figure 2.

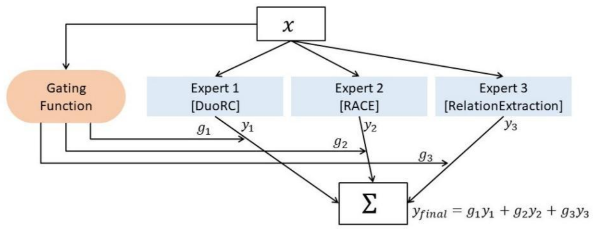


Figure 2: Mixture of Experts structure

For our specific implementation, we first converted the examples into token embeddings, which is then fed into the gating function. The gating function was designed to be a two-layered bidirectional LSTM followed by a linear layer and a softmax function, which transforms the outputs of the LSTM model into probabilities. The final predictions of the MOE model is the weighted combination of predictions for the three models, as shown by Equation 1:

$$p(y_i|\boldsymbol{x}_i, \theta) = \sum_k p(z_i = k|\boldsymbol{x}_i, \theta)p(y_i|\boldsymbol{x}_i, z_i = k, \theta) \tag{1}$$

Here, $p(z_i = k|\boldsymbol{x}_i, \theta)$ represents the probability given by the gating function to expert k and $p(y_i|\boldsymbol{x}_i, z_i = k, \theta)$ represents the probability for the ground truth to be the truth provided by expert k.

For model optimization, the total loss of the MoE model was calculated by aggregating the negative log likelihood loss of both the start and end of the answers based on the final predictions. The gating LSTM, the linear layer and the three models are optimized simultaneously.

3.4 Mixture of Experts with Data Augmentation

Finally, to see if utilizing the capability of the MoE model to specialize in different distributions together with the augmented data could facilitate the model to better learn the invariant features of the out-of-domain datasets, we used the mixture-of-expert models we built in the previous section on the out-of-domain datasets along with our augmented data in a series of experiments.

4 Experiments

4.1 Data

We used three in-domain datasets, including Natural Questions, NewsQA and SQuAD, for training, and three out-of-domain datasets (RelationExtraction, DuoRC, RACE) for testing. Except for the baseline, a small number of samples from the out-of-domain datasets are also included during training.

4.2 Evaluation method

We used EM score, which measures the percentage of predicted answers that exactly match any one of the ground truth answers, and F1 score, which is calculated based on the precision and recall scores, as our evaluation metrics.

4.3 Experimental details

4.3.1 Baseline Model

For the baseline model finetuned with in-domain datasets, we used the provided configurations. Specifically, we used a batch size of 16 and 3 total number of epochs. The seed was 42, and the learning rate 3e-5. We evaluated the model and saved the checkpoints every 2000 batches. Based on this baseline model, we used the same configurations to further finetune the model on the few out-of-domain training examples we have and obtained baseline model 2.

4.3.2 Hyperparameter Tuning

Given the small size of the out-of-domain data, we further tuned our hyperparameters based on the baseline models to see if we could achieve better results. Specifically, we ran two rounds of hyperparameter tuning. In the first round, we used a random grid to test 20 sets of hyperparameters. Based on the results, we narrowed down the scope for the learning rate and number of epochs, and ran another round of hyperparameter tuning for 20 sets through a random grid and identified the top 5 sets of hyperparameters from the two rounds. The specific configurations are the following:

Hyperparameters	**Round 1**	**Round 2**
num_epochs	[3, 5, 8, 10]	[3, 5, 8]
batch_size	[5, 10, 16, 20]	[5, 10, 16, 20]
lr	[3e-3, 3e-4, 3e-5, 3e-6]	[3e-5, 3e-6]
eval_every	[5, 7, 10, 15]	[5, 7, 10, 15]

Table 1: Hyperparameters

4.3.3 Data Augmentation

For data augmentation, all techniques were performed with an alpha of 0.05, meaning that for each sample, 5% of the context was perturbed based on their context length (the total number of words). For model finetuning, we used the best performing set of hyperparameters and adjusted the eval_every parameter to maintain the same evaluation frequency for all of the experiment sets. Using baseline model 1 as a starting point, we ran a total of 8 experiments: experiment 1-4 use the original OOD datasets combined with the data augmented with each of the SR, RS, RI, RD techniques individually. Experiment 5 uses the original OOD datasets combined with out-of-domain data augmented by all of the methods. After seeing no significant improvement compared to experiment 1-4, we carried out experiments 6-8, where we used the original OOD datasets, RD augmented data, along with one of RS, RI, SR augmented datasets.

4.3.4 Mixture of Experts

For the mixture-of-experts method, we first transformed our inputs through the tokenizer from the DistilBERT model. Based on the vocabulary size, we used the `Embedding` class from the `torch.nn` module to obtain the text embeddings. To be consistent with the DistilBERT model, we used an embedding size of 768. For the Bi-directional LSTM, due to virtual machine cuda memory constraint, we used a hidden size of 3. For the softmax function, we introduced a temperature to control the probability distribution over the experts and had the default set to 1. During learning, we optimized the overall model with one Adam optimizer, and give the option to vary the learning rate for the gating function with the default set to 3e-5. For the overall configurations, we used the best performing set of hyperparameters but adjusted the batch size to 10 due to cuda memory constraints.

4.3.5 Mixture of Experts with Data Augmentation

We used the same architecture and followed the same procedures as in the previous sub-section for this set of experiments. Specifically, we trained the models for 3 epochs with a learning rate of 3e-05, a batch size of 10 and set the evaluation frequency based on the overall size of the out-of-domain training data. The temperature was set to 1 and the learning rate for the gating function was 3e-5. We ran a total of 5 experiments, one for each augmented dataset along with the original out-of-domain training samples, and one for all of the augmented datasets combined.

5 Results

5.1 Baseline

As a baseline, we finetuned a pre-trained DistilBERT model with Natural Questions, NewsQA and SQuAD datasets. We evaluated the model performance on RelationExtraction, DuoRC, RACE datasets and obtained an F1 score of 47.10 and EM score of 31.68. By further finetuning the model with the few out-of-domain training samples we have using the same configurations, we obtained baseline model 2, which has slightly better results: the F1 score is 47.41 and the EM score is 32.20.

5.2 Hyperparameter Tuning

During our first round of hyperparameters tuning, we found learning rates of 3e-3 and 3e-4 perform particularly worse than others. After inspecting the negative log likelihood, F1, and EM plots for the training process, we also found that training for 3 epochs was enough for the model to converge, but training for up to 8 epochs could potentially lead to more stable performance. After discarding the two learning rate values and narrowing the scope for the number of epochs, we ran another round of random search. We identified the Top 5 best performing sets of hyperparameters in Table 2, which were ranked by their F1 scores.

batch_size	eval_every	lr	num_epochs	F1	EM
16	7	3.00E-05	8	50.75	35.34
16	10	3.00E-05	3	50.74	36.91
10	7	3.00E-05	3	50.52	35.60
5	15	3.00E-05	3	50.42	36.91
5	7	3.00E-05	8	50.01	35.86

Table 2: Hyperparameters Tuning on OOD baseline model - Top 5

Because the second set of hyperparameters achieves a similar level of F1 but higher EM score with much fewer epochs than the first set of hyperparameters, we decided to use this as the default setting for our data augmentation and mixture-of-experts experiments. These results are what we expected: rigorous hyperparameters tuning can often lead to superior performance. Performance should stabilize once training reaches convergence. Training beyond that could lead to overfitting.

5.3 Data Augmentation

	Models Finetuned with Different Datasets	F1	EM
1	OOD-train-datasets + RD augmented datasets	52.71	35.60
2	OOD-train-datasets + SR augmented datasets	52.45	34.55
3	OOD-train-datasets + RI augmented datasets	52.21	34.55
4	OOD-train-datasets + RS augmented datasets	51.99	35.08
5	OOD-train-datasets + SR,RD,RI,RS augmented datasets	52.81	31.41

Table 3: Models finetuned with Augmented Data

Table 3 shows the results of the models finetuned on the original OOD data along with the augmented data. When only a single EDA technique is applied (experiment 1-4), the models all have moderate improvement for F1 score over their corresponding baseline. However, decrease in scores is also observed. The results generally match our expectations: introducing some variations to the data improves F1 score through reducing brittleness of the model in out-of-domain contexts, but also makes the answers less consistent with the standard format, thus decreasing the EM score. On the other hand, the result from experiment 5 did not match our expectation. We hypothesized that adding more data augmented by different methods would further improve the results. However, experiment 5 shows that the model finetuned with all-method augmented datasets actually has a much lower EM score (31.41) despite a slightly improved F1 score (52.81). In order to understand where the drop in performance comes from, we further performed experiments 6 to 8 (See appendix A.3). Seeing that the results are quite similar to experiment 2-4, we theorize that the drop of EM score for trial 5 may be caused by overfitting since the model was trained with the most data, many of which have similar context and question with each other.

5.4 Mixture of Experts

5.4.1 The Naive Approach

The Naive MoE model without the gating function achieved an F1 score of 53.64 and an EM score of 34.82, which are both higher than the scores from a single model with the same hyperparameters. Such results are what we expected: since we trained three separate models, each on a different dataset, the model can achieve better results by specializing in one of distributions. Our hypothesis was confirmed when we checked the performance of each individual model: the F1 and EM score can reach as high as 78.37 and 46.07 respectively for the Relation Extraction dataset, while falling as low as 39.33 and 24.20 for the race dataset. In comparison, the scores based on a single model are F1 of 72.08 and EM of 51.56 for the relation extraction dataset and F1 of 35.89 and EM of 21.88 for the race dataset. In an ideal situation where we know the evaluation data distribution and do not need the model to be a generalist, this approach would perform well.

5.4.2 The Main Approach

Trial	gating_lr	T	F1	EM	Trial	gating_lr	T	F1	EM
1	3.00E-05	1	50.01	35.86	4	3.00E-04	1	50.27	35.86
2	3.00E-05	0.1	49.33	35.60	5	3.00E-06	1	49.96	36.13
3	3.00E-05	10	50.28	35.60	6	3.00E-04	10	49.66	35.08

Table 4: Mixture-of-Experts Models

Table 4 shows the performance of the mixture-of-expert approaches with different temperature and gating function learning rates. Trial 1 is the plain vanilla version of our MoE model, which serves as a control. Trial 2-3 have the same gating function learning rate with the control model but different temperatures, while Trial 4-5 have the same temperature as the control model but different gating function learning rates. Trial 6 uses a combination of different temperature and learning rate. Compared to the base model with the same hyperparameters, Trial 1 sees slight but non-significant improvement. This goes against our expectation. The lukewarm performance could be due to the small out-of-domain sample size we have, which may not be enough to train a good gating function.

In future experiments, we would want to try out a different gating function such as MLP or slightly increase the training size and compare the results. Compared to the control trial (F1: 50.01, EM: 35.86), we do not see significant improvement by modifying the temperature and the learning rate for the gating function. This again goes against our expectations. We originally thought the performance will be better when we decrease the temperature, which would assign a larger probability to the most probable expert and lead the experts to learn specialized behavior. However, our results suggest that the probability distribution could already be skewed. Visual inspection of a few matrices confirmed our suspicion. As a result, using a small fixed temperature will not change the performance much. In future experiments, performance of significantly lower and higher temperatures could be evaluated.

5.5 Mixture of Experts with Data Augmentation

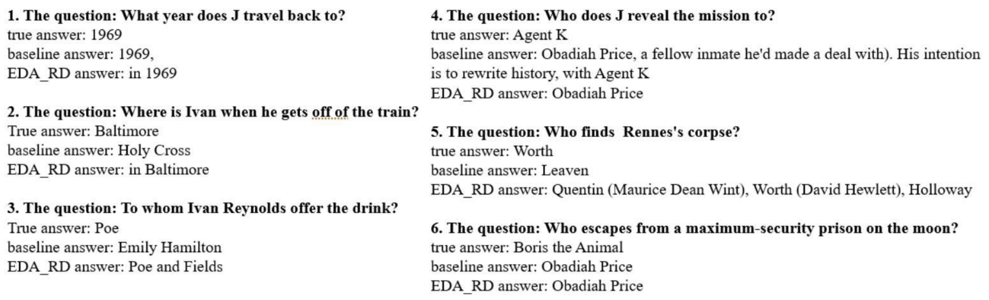
	Data	**F1**	**EM**		**Data**	**F1**	**EM**
1	OOD	50.01	35.86	4	OOD + RS	52.50	35.34
2	OOD + RD	52.74	36.13	5	OOD+ SR	52.35	33.77
3	OOD + RI	54.18	36.39	6	OOD + All	51.23	29.32

Table 5: Mixture-of-Experts Models with Augmented Data

Table 5 shows the results of the mixture-of-expert models with augmented data. Model 2-5, which are finetuned with the original OOD and one of the augmented datasets show significant improvement over the performance of model 1, which is finetuned with only the original OOD datasets. Random insertion alone shows the most improvement. When evaluated on the test leaderboard, it achieved an F1 score of 62.062 and an EM score of 42.317. Our hypothesis for the good performance of the method is that random insertion gives more variations in answer positions and thus allows different experts of the MoE model to learn patterns not correlated with the answer positions. This advantage may not have been significant for EDA since signals from other datasets may distract the attention of the model. With multiple experts, this advantage becomes more significant. However, we will need more trials with similar methods such as random insertion of irrelevant sentences to confirm our hypothesis. For Model 6, similar to the results in Section 5.3, the F1 score only increased slightly, while the EM score drops significantly, which again could be due to overfitting.

6 Analysis

In order to understand the differences in scores behind our models, we visually examined the outputs of all of our approaches (See appendix A.4 for additional data). As a demonstration of our findings, we created Figure 3, which shows six representative samples that include the question, the ground truth, and the predicted answers by baseline model 1 and the model trained on OOD+RD datasets.



1. The question: What year does J travel back to?
true answer: 1969
baseline answer: 1969,
EDA_RD answer: in 1969

2. The question: Where is Ivan when he gets off of the train?
True answer: Baltimore
baseline answer: Holy Cross
EDA_RD answer: in Baltimore

3. The question: To whom Ivan Reynolds offer the drink?
True answer: Poe
baseline answer: Emily Hamilton
EDA_RD answer: Poe and Fields

4. The question: Who does J reveal the mission to?
true answer: Agent K
baseline answer: Obadiah Price, a fellow inmate he'd made a deal with). His intention is to rewrite history, with Agent K
EDA_RD answer: Obadiah Price

5. The question: Who finds Rennes's corpse?
true answer: Worth
baseline answer: Leaven
EDA_RD answer: Quentin (Maurice Dean Wint), Worth (David Hewlett), Holloway

6. The question: Who escapes from a maximum-security prison on the moon?
true answer: Boris the Animal
baseline answer: Obadiah Price
EDA_RD answer: Obadiah Price

Figure 3: Sample Question-Answer Pairs

After visually inspecting the answers, we divided them into four groups: first, the answer exactly matches the ground truth. The percentage of samples belonging to this category can be directly seen through the EM score, so we do not go into detail here for this group. Second, the answer is slightly different from the ground truth, but the meaning is the same. The answers in the first example of Figure 3 is representative of this category: the predicted answers are only different from the ground truth due to punctuation marks or preposition. Third, the answer contains not only the ground truth but also other irrelevant words. The baseline answer in the fourth examples demonstrates this point. Finally, the answer could be completely wrong, which is demonstrated by answers in question 6.

What we found in our analysis is that, the increase in F1 score for the EDA methods compared to the baseline could be largely explained by the third category. After noticing the prevalent presence of long answers containing the ground truth (like the baseline answer in Figure 3 question 4), we hypothesized that the worse performing baseline model could be giving a larger percentage of predictions of the third category. We thus analyzed and compared the average word count in predicted answers by models and the ground truth answers, the result of which is shown by Figure 4 and Table 6.

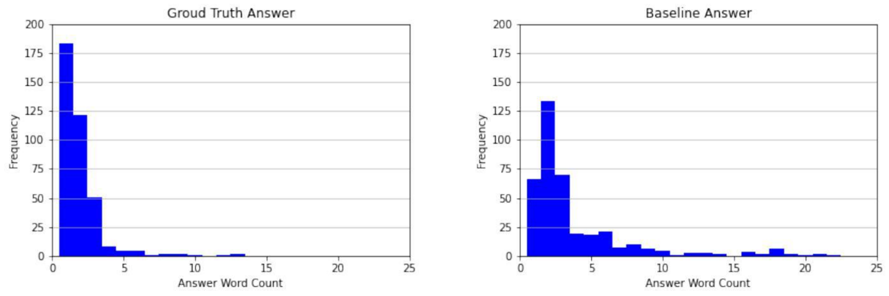


Figure 4: Histogram of word count in answers predicted by different models

	Truth	Baseline	EDA_RD	EDA_RI	MoE_RI
Average Word Count	1.97	3.95	3.52	3.88	3.33

Table 6: Average Word Count

From the statistics and the histograms, we can see that the average word count of answers predicted by each model is about twice as much as the word count of the ground truth answers, and the worse performing baseline model indeed has a higher average word count for their answer predictions than the corresponding EDA predictions. This pattern is also reflected in the MoE models, as the best performing MoE-RI model has the lowest average word count among all models. More detailed distribution of word counts is included in Appedix A.4. Seeing that the model predicts the start and end positions of an answer to form an answer span, we suspect that the reason behind the model giving long answers is that the model may give a wrong token similar to the true end token a slightly higher probability. A qualitative, non-comprehensive analysis of the predictions supports our conjecture (for example, the baseline answer in question 4 starts and ends with nouns representing persons). A more rigorous linguistic analysis would be needed to draw a definitive conclusion.

7 Conclusion

This project implements and studies the efficacy of data augmentation techniques and the mixture-of-experts approach in improving the robustness of QA systems on out-of-domain data. Through a series of experiments, we demonstrated the effectiveness of the two methods and showed that their combination strengthens each other's performance. Specifically, using random-insertion augmented datasets with the MoE model helped us achieve 15.03% and 14.87% gains for the F1 and EM scores on the dev sets, and reach an F1 score of 62.062 and an EM score of 42.317 on the test leaderboard.

The intermediate results of the project also highlight some salient issues and limitations of our work. Through hyperparameters tuning, we found that having an optimal set of hyperparameters can non-trivially boost the model performance, giving us dev F1 scores in the low 50s. While demonstrating the importance of hyperparameter tuning, these results also raise some challenges for us to reach definitive conclusions on whether some of the methods actaully boost system performance, seeing that the scores are close for the size of our evaluation datasets. To confirm that the improvement was actually made by our methods, we would like to run multiple experiments for each of the methods, obtain the average score, and also bootstrap the samples to obtain a confidence interval for all of the methods. We leave these as our future work.

References

[1] Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. *CoRR*, abs/1611.01604, 2016.

[2] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016.

[3] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. *CoRR*, abs/1608.07905, 2016.

[4] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[5] Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Association for Computational Linguistics (ACL)*, 2020.

[6] Robin Jia and Percy Liang. Adversarial examples for evaluating reading comprehension systems. In *EMNLPà*, 2017.

[7] Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Association for Computational Linguistics (ACL)*, 2018.

[8] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *arXiv preprint arXiv:1901.11196*, 2019.

[9] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Semantically equivalent adversarial rules for debugging nlp models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[10] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale, 2018.

[11] R. Jacobs, S. Nowlan Michael I. Jordan, and Geoffrey E. Hinton. Adaptive mixtures of local experts. In *Neural Computation*, 1991.

[12] S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. *IEEE Transactions on Neural Networks and Learning Systems*, 23(8):1177–1193, 2012.

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

A Appendix

A.1 Baseline

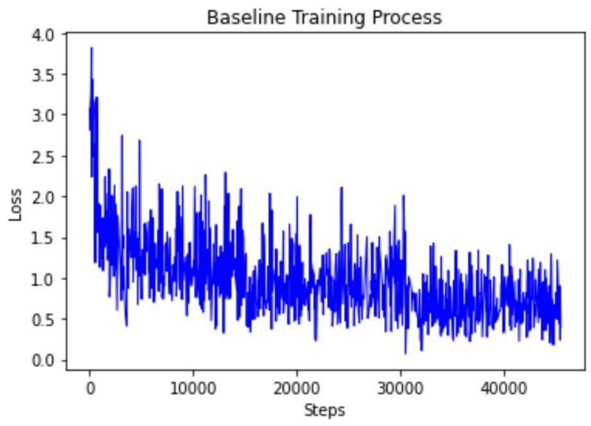
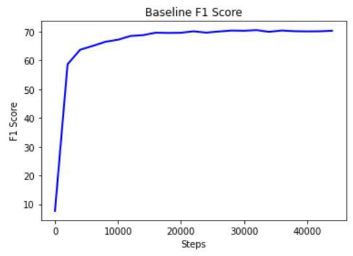
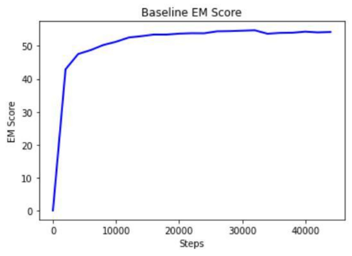


Figure 5: Negative Log Likelihood loss for in-domain finetuning



(a) F1 score for in-domain finetuning



(b) EM score for in-domain finetuning

Baseline	**F1**	**EM**
Baseline Model 1 with only in-domain datasets	47.10	31.68
Baseline Model 2 with OOD-train-datasets, eval-every 200	47.41	32.20

Table 7: Baseline Models: Pre-trained DistilBERT finetuned with our training datasets

A.2 Hyperparameter Tuning

rank	batch_size	eval_every	lr	num_epochs	F1	EM
1	16	7	3.00E-05	8	50.75	35.34
2	5	7	3.00E-05	8	50.01	35.86
3	5	15	3.00E-05	8	49.71	36.91
4	20	7	3.00E-05	5	49.66	35.86
5	5	5	3.00E-06	8	48.8	34.82
6	20	10	3.00E-06	3	47.4	31.94
7	7	7	3.00E-06	3	47.3	31.94
8	16	7	3.00E-04	7	45.96	31.41
9	10	15	3.00E-04	8	45.55	31.41
10	10	10	3.00E-04	10	45.18	31.41
11	16	7	3.00E-04	8	44.93	30.63
12	5	7	3.00E-04	10	44.59	28.8
13	20	7	3.00E-04	3	44.1	30.37
14	10	7	3.00E-04	10	43.84	29.32
15	16	7	3.00E-04	10	42.96	27.75
16	10	5	3.00E-04	3	42.32	27.49
17	16	7	3.00E-03	10	11.42	3.14
18	5	10	3.00E-03	8	9.05	0.52
19	20	10	3.00E-03	10	8.25	1.05
20	10	15	3.00E-03	8	7.21	0.52

Table 8: Hyperparameter Tuning Round 1

rank	batch_size	eval_every	lr	num_epochs	F1	EM
1	16	10	3.00E-05	3	50.74	36.91
2	10	7	3.00E-05	3	50.52	35.6
3	5	15	3.00E-05	3	50.42	36.91
4	5	5	3.00E-05	3	49.68	33.77
5	10	15	3.00E-05	3	49.62	35.86
6	5	10	3.00E-05	3	49.6	35.6
7	20	7	3.00E-05	3	49.32	34.29
8	16	15	3.00E-05	3	49.21	35.34
9	20	5	3.00E-05	8	49.16	34.55
10	16	10	3.00E-05	5	49.09	35.86
11	5	15	3.00E-05	8	48.98	33.77
12	10	10	3.00E-05	3	48.84	33.77
13	5	10	3.00E-05	3	48.67	34.82
14	16	15	3.00E-05	5	48.32	33.77
15	5	10	3.00E-06	5	47.55	32.72
16	5	5	3.00E-06	3	47.4	31.94
17	5	7	3.00E-06	3	47.34	31.94
18	20	10	3.00E-06	8	47.33	31.94
19	20	15	3.00E-06	5	47.1	31.68
20	16	10	3.00E-06	3	47.1	31.68

Table 9: Hyperparameter Tuning Round 2

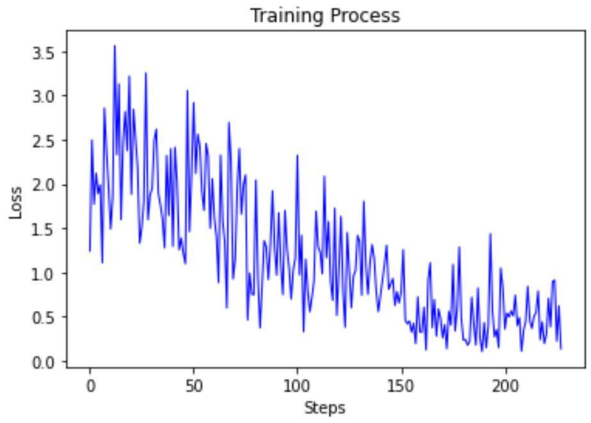
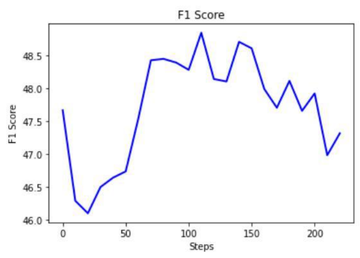
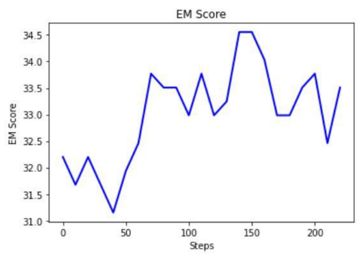


Figure 6: Negative Log Likelihood loss for finetuning with the first set of hyperparameters in round 2



(a) F1 score for finetuning with the first set of hyperparameters in round 2



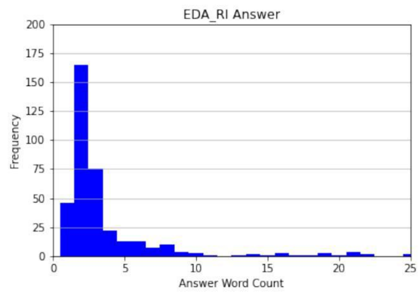
(b) EM score for finetuning with the first set of hyperparameters in round 2

A.3 Data Augmentation Experiments

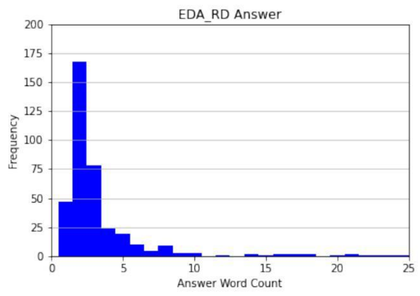
	Models Finetuned with Different Datasets	**F1**	**EM**
1	OOD-train-datasets + RD augmented datasets	52.71	35.60
2	OOD-train-datasets + SR augmented datasets	52.45	34.55
3	OOD-train-datasets + RI augmented datasets	52.21	34.55
4	OOD-train-datasets + RS augmented datasets	51.99	35.08
5	OOD-train-datasets + All augmented datasets	52.81	31.41
6	OOD-train-datasets + RD, RI augmented datasets	51.97	33.20
7	OOD-train-datasets + RD, SR augmented datasets	51.93	33.98
8	OOD-train-datasets + RD, RS augmented datasets	52.96	33.89

Table 10: Augmented Models.

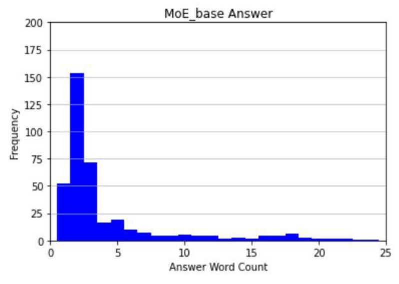
A.4 Average Word Count for MoE Models with EDA



(c) EDA_RI Answer

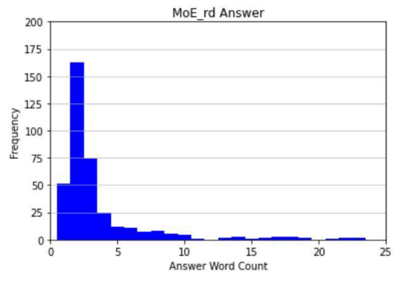


(d) EDA_RD Answer



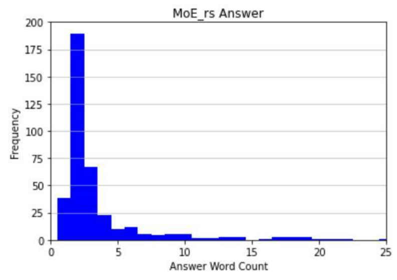
```
average answer length is 4.2408376963350785
```

(e) MoE_base Answer



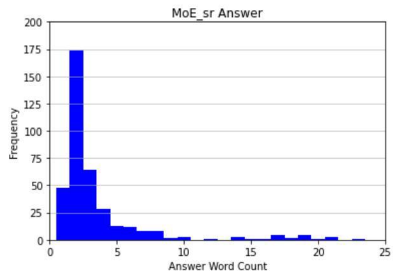
```
average answer length is 3.6727748691099475
```

(f) MoE_RD Answer



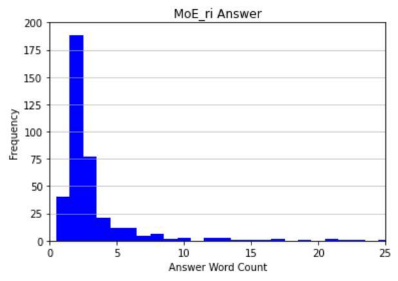
```
average answer length is 3.6282722513089007
```

(g) MoE_RS Answer



```
average answer length is 3.6727748691099475
```

(h) MoE_SR Answer



```
average answer length is 3.329842931937173
```

(i) MoE_RI Answer

Figure 7: Histogram of word count in answers predicted by MoE models with EDA

	Truth	Baseline	EDA_RD	EDA_RI	MoE_RI
Average Word Count	1.97	3.95	3.52	3.88	3.33
	Truth	MoE	MoE_RD	MoE_RS	MoE_SR
Average Word Count	1.97	4.24	3.67	3.62	3.67

Table 11: Average Word Count