DAM-Net: Robust QA System with Data Augmentation and Multitask Learning

Stanford CS224N Default (RobustQA) Project

Siyun Li
Stanford University
lisiyun@stanford.edu

Xi Yan
Stanford University
xiyan@stanford.edu

Yige Liu
Stanford University
yigel@stanford.edu

Abstract

While a plentitude of models has shown on-par performance with humans on question answering (QA) given context paragraph, several works have shown that they generalize poorly on datasets that are dissimilar to their training distributions. In this project, we aim to train and fine-tune a robust QA model that can achieve strong performance even on test examples drawn from out-of-domain distributions. Specifically, we perform data augmentation on our training data, expand training with the auxiliary task (i.e. fill-in-the-blank), and utilize multi-domain training and additional fine-tuning. We further combine all three approaches using ensemble, which offers additional performance boost. Our best model achieves EM/F1 score of **40.58/55.68** on the validation set and EM/F1 score of **45.14/62.16** on the test set, ranking **top 1** on both the validation and test leaderboards (as of Mar 16, 2021). [1]

1 Introduction

With the advent of large-scale pre-trained language models such as BERT [1], we have seen tremendous progress towards building machines that can truly understand and reason with human language. Using transfer learning, these models have demonstrated on-par performance with humans on a large variety of natural language understanding tasks [1, 2, 3]. However, several works have shown that these models generalize poorly beyond their training distributions [4, 5]. This is problematic since, in the real world, these models are often used in cases where queries come from domains that are different from their training data. Thus, it is particularly important to build robust systems that can adapt to unseen domains while still achieving strong performance.

In this project, we focus on the task of closed-domain question answering (QA) since it is commonly used to measure how well a computer system understands the human language [6]. In the QA task, the model is given input pairs consisting of a context paragraph and a question related to the paragraph. The goal of the model is to select the span of text in the paragraph that answers the question. Specifically, we build a DistilBERT-based [7] question answering system that works well on out-of-domain datasets. We perform data augmentation on our training dataset and train the QA model jointly with an masked language model as an auxiliary task. We also perform additional fine-tuning on our trained models using few out-of-domain examples and utilize ensemble to obtain our best model. Overall, our final submitted model achieves +6.03/+6.26 EM/F1 over the baseline on the out-of-domain validation set.

2 Related Work

In recent years, there has been increasing attention in investigating the robustness of NLP systems to out-of-domain data [4, 6]. Some methods include few-shot learning [8], domain adversarial training [9], and data augmentation [10]. Meanwhile, a multitude of QA benchmark datasets has

[1] **Key Information to include:** We have no external collaborators, mentor, and we are not sharing the project.

been developed. Collectively, they provide improved coverage in multiple domains, ranging from Wikipedia such as SQuAD[11], NaturalQuestions[12], RelationExtraction[13], to news article [14], movie reviews [15] , and examinations [16]. [6] does a survey for addressing the generalization capabilities of reading comprehension systems. Our work tackles a similar task in which we aim to generalize to out-of-domain datasets.

In our work, we investigate several techniques for improving robustness of language models including data augmentation [17, 18], task adaptive fine-tuning [19], and ensemble [20]. Most related to our work, D-Net [21] uses ensemble of different BERT-base models (XLNet [2], ERNIE 2.0 [3], BERT [1]), and multi-task training [19]. Unlike their work, we focus on the smaller DistilBERT[7] model with fewer training examples and additionally perform data augmentation. [17] proposes several easy data augmentation techniques for text classification including synonym replacement (SR) and random insertion (RI). In our work, we show that these techniques also work well for the task of question answering. Inspired by [10, 22], we also experiment with back translation data augmentation technique for question answering.

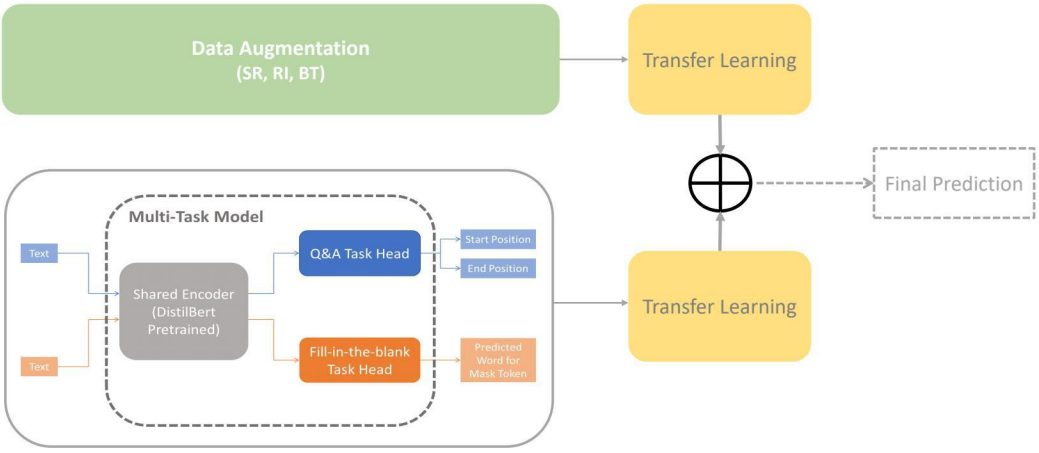

Figure 1: Overview of our DAM-Net. We use two techniques (1) data augmentation and (2) multitask learning. We then perform transfer learning on the resultant models and ensemble their outputs to produce the final prediction.

3 Approach

In this section, we first describe our baseline approach in Section 3.1. We then provide details on our original work: transfer learning (TL) setup in Section 3.2, data augmentation (DA) techniques in Section 3.3, multitask training (MT) in Section 3.4, and ensemble (EN) in Section 3.5. Figure 1 gives an overview of our approach.

3.1 Baseline

We use the provided baseline in the project handout [23]. Specifically, the baseline model is a pre-trained DistilBERT [7] model with a single linear layer as classification head for QA. The model is initialized with the default configuration and finetuned on `indomain-train`. Please refer to the handout and starter code for more details.

3.2 Transfer Learning

We first perform preliminary experiments with different ways to incorporate the given `indomain-train` and `oodomain-train` datasets so that our model can generalize to out-of-domain test sets. To this end, we trained models using (1) **multi-domain training** [4] where the model was trained on the union of `indomain-train` and `oodomain-train` on a total of 6 datasets; (2) **additional finetuning** using the `oodomain-train` on top of a trained QA model; and (3) a combination of (1) and (2).

3.3 Data Augmentation

Data augmentation techniques have proven to boost performance on various NLP tasks [17]. We use several data augmentation techniques on both the context paragraph and the query sentence for the out-of-domain datasets.

Synonym Replacement (SR): We implement a Synonym Replacement (SR) operation for paragraphs in our training data. We randomly choose 10% of words in the paragraph that are not stop words and replace these words with one of its synonyms chosen at random. We implement our approach using the `SynonymAug` in the nlpaug [24] package with synonyms from WordNet [25], based on techniques "Random Swap" and "Stopword Dropout" described in [26]. We show a snippet comparing the original and the augmented paragraph from the RACE dataset with replaced synonyms highlighted.

Original Paragraph

You love Jay Chou's songs and you can sing some quite well. So you make a video of your performance and post it online for your friends to see. But what if this led to something beyond your wildest imagination–a career in music? Canadian teenager Justin Bieber, 16, has just had the magical experience: He posted homemade videos of his versions of songs by American singer Chris Brown online for his relatives.

Augmented Paragraph

You love Jay Cabbage's songs and you can blab some quite well. So you take a leak a video of your performance and post it online for your friends to see. But what if this led to something beyond your wildest imagination–a calling in music? Canadian teenager Justin Bieber, 16, has just had the wizardly experience: He posted homemade videos of his versions of songs by American vocaliser Chris Brown online for his relatives.

Random Insertion (RI): We also augment the context paragraph using the Random Insertion (RI) operation as described in [17]. Here, we pick a random word in the paragraph that is not a stop word, and find a random synonym according to TF-IDF calculation [18]. Then, we insert that synonym into a random position in the sentence. More specifically, We implement this using `TfIDfAug` from nlpaug, with the insertion operation done on 10% of the words. The following shows an snippet comparing the original and the augmented paragraph from RelationExtraction dataset with inserted word highlighted.

Original Paragraph

Ray Eberle died of a heart attack in Douglasville, Georgia on August 25, 1979, aged 60.

Augmented Paragraph

- Ray Eberle died of a heart attack in Douglasville, Coast Georgia on August 25, 1979, aged 60.
- Ray Eberle finally died of a heart attack in Douglasville, Georgia on August 25, 1979, aged 60.

Back Translation (BT): We consider a back translation technique where we paraphrase the examples by translating the original sentences from English to another language and then back, as introduced by [22]. Back translation is performed on each question rather than context as this alleviates the problem that the context may be paraphrased such that the original answer span is no longer in the back-translated context. We implement this using the Google Cloud Translate API [2], and use three different intermediate languages: French, German, and Italian.

Original Question

What room does Kitty volunteer in when they arrive in China?

[2] `https://github.com/googleapis/python-translate`

Augmented Questions

- **English**→**French**→**English**: What room does Kitty volunteer in when she arrives in China?
- **English**→**German**→**English**: In which room does Kitty volunteer when she arrives in China?
- **English**→**Italian**→**English**: Which room does Kitty volunteer in when they arrive in China?

3.4 Multitask QA + fill-in-the-blank Head:

Previous work has demonstrated that adding auxiliary tasks in the fine-tuning phase along with QA task can improve the generalization of QA models [21]. Inspired by Multi-Task Learning (MTL) [27] on supervised dataset for NLP models, we incorporate the unsupervised, fill-in-the-blank task as an auxiliary task. Given an input sentence, we randomly replace 15% of the words with the `<mask>` token at 80% probability, with random word at 10%, and no replacement at 10% [1]. The classification head for this auxiliary task predicts the word for each `<mask>` token. In the following sections, we refer to this classification head as the masked language model (MLM) task head.

Figure 1 shows our multi-task model. We use a single pre-trained DistilBERT model as our shared encoder and add two separate classification heads for QA and fill-in-the-blank tasks [3]. During training, we sample batches of data from either task with equal probability and run a forward pass through the shared encoder and the corresponding task head. When we back-propagate the gradients from each head, the shared encoder will also have its weights updated. This ensures that we can train the encoder jointly on both tasks. For both task-specific heads, we compute the cross-entropy loss between predictions and labels (or ground truth before masking in the fill-in-the-blank task). During evaluation, we only use the QA dataset and task head.

3.5 Ensemble

Ensemble has proven to be an effective technique in boosting the performance as well as the robustness of ML models. We combine our models using a simple ensemble scheme by taking the majority vote of all model predictions. We show that this yields around +2/+3 EM/F1 boost over single models on the out-of-domain validation set.

4 Experiments

4.1 Data

In this project, we use the six provided datasets, three of which are in-domain and others are out-of-domain. Detail statistics on these datasets can be found in the project handout [23]. For data augmentation, we create a set of augmented data on `oodomain-train` using the approachs described in Section 3.3. We also use an additional dataset, `ms-macro` v1.1[28], for training the MLM head in our multi-task model. This dataset is pre-processed and directly obtained through Hugging Face's dataset API. Due to RAM constraints, we only use the first 500,000 passages in the training set. It should be noted that though this dataset is labeled, we only use it for the unsupervised, fill-in-the-blank task.

4.2 Evaluation method

We evaluate our model using two metrics: Exact Match (EM) score and F1 score. EM score is a strictly binary (0/1) measure of whether the model selects the exact same answer (i.e. sequence of words) as the ground truth. F1 score combines precision and recall accuracies of the model using the following equation:

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{1}$$

where TP/FP is true/false positive, and FN is false negative.

Note that for our multi-task model, we only evaluate the performance of the QA head using these two metrics. A model is considered better if it results in higher F1 score for the QA task.

[3] For data pre-processing, we adopted code from `https://github.com/google-research/bert`

4.3 Experimental details

For all approaches, we implemented our own code based on the provided starter code[4] in PyTorch and the HuggingFace's transformer API[5]. For synonym replacement and random insertion operations, we use the `nlpaug` package [24]. For back translation, we use the Google Cloud Translation API [6].

In our experiments, we train models using all our approaches described in Section 3. #1-4 are our transfer learning (TL) experiments, #5-7 are our data augmentation (DA) models, and #8-10 are our multi-task (MT) models. All our models are trained on Google Cloud Platform (GCP) VM with NVIDIA Tesla V100 (16G) and 78GB CPU RAM. We leverage Microsoft Azure VM with NVIDIA K80 (11G) and 56 GB CPU RAM for development and debugging. All models are trained with a batch size of 16, and learning rate of 3e-05. Table 1 reports the different hyperparameter configurations for each model.

#	Training Data	Epochs	Max Length	Eval-Every	Val Data
1 TL	ind	3	384	2000	`in-val`
2 TL	ind+ood	10	384	2000	`in-val`
3 TL	#1 (+ ood ft)	10	384	100	`ood-val`
4 TL	#2 (+ ood ft)	10	384	10	`ood-val`
5 DA	ind+ood+ood-sr-aug	10	512	2000	`ood-val`
6 DA	ind+ood+ood-back-aug	10	384	2000	`ood-val`
7 DA	ind+ood+ood-all-aug	10	384	2000	`ood-val`
8 MT	ind	3	384	2000	`in-val`
9 MT	ind+ms-marco	3	384	2000	`in-val`
10 MT	ind+{ind+ms-marco}	3	384	2000	`in-val`
11 MT	#8 (+ ood ft)	15	384	20	`ood-val`
12 MT	#9 (+ ood ft)	15	384	20	`ood-val`
13 MT	#10 (+ ood ft)	15	384	20	`ood-val`

Table 1: Hyperparameter settings for different models. ind stands for `indomain-train`, ood stands for `oodomain-train`, ood-sr-aug stands for augmented data using synonym replacement, ood-back-aug stands for augmented data using back translation, and ood-all-aug stands for augmented data using synonym replacement, random insertion, and back translation.

4.4 Results

Table 2 shows the main results of all our approaches. From the results in #1-4, we have observed that multi-domain training improves generalization performance as was observed by [10, 4]. We also find that an additional round of finetuning on `oodomain-train` offers a performance boost on the out-of-domain validation dataset, likely due to the fact that this allows the model to learn features in out-of-domain distribution. Training with additional augmented data (#5-7) offers around +2/+1 in EM/F1 boost over the baseline on the `oodomain-val`. This aligns with our initial assumption that adding data in the out-of-domain distribution will improve performance in out-of-domain.

However, while additional fine-tuning and adding out-of-domain augmented data improves the `oodomain-val` performance, we also see a slight decrease in the `indomain-val` set. We suspect that this is because these technique skews the model to learn better representation on the out-of-domain distribution but forgets some of the representations learned on in-domain data. This does not happen with MultiDomain (#2) training which shows improvement on both `indomain-val`, `oodomain-val` over the baseline.

To our surprise, we do not observe much improvement from adding the fill-in-the-blank auxiliary task. We suspect that this is due to the lack of data for MLM task head, since when we leverage the unsupervised `ms-macro` dataset, the model performs better than only using in-domain datasets when training with MLM. Nonetheless, when we perform an additional fine-tuning on top of our

[4] `https://github.com/MurtyShikhar/robustqa`

[5] `https://huggingface.co/transformers/model_doc/distilbert.html`

[6] `https://github.com/googleapis/python-translate`

#	Model Description	+ F.t.	indomain-val		oodomain-val	
			F1	**EM**	**F1**	**EM**
1 TL	Baseline	-	70.66	55.09	49.42	34.55
2 TL	MultiDomain	-	70.85	54.68	51.26	35.60
3 TL	Add.Finetune on #1	✓	64.98	49.23	51.35	36.13
4 TL	Add.Finetune on #2	✓	69.37	53.16	52.92	38.48
5 DA	DataAug SynReplace	-	69.03	52.45	51.88	36.13
6 DA	DataAug BackTrans	-	69.02	53.22	50.84	36.91
7 DA	DataAug SynReplace+RandIns+BackTrans	-	69.18	53.32	51.42	37.17
8 MT	MultiTask	-	70.02	53.87	44.57	30.63
9 MT	MultiTask, `ms-marco` for MLM	-	69.64	53.96	48.16	33.51
10 MT	MultiTask, `ind + ms-marco` for MLM	-	69.96	53.60	46.97	31.68
11 MT	Add.Finetune on #9	✓	64.98	48.86	49.87	33.51
12 MT	Add.Finetune on #10	✓	65.02	49.18	50.14	34.82
13 MT	Add.Finetune on #11	✓	65.86	49.46	51.05	36.91

Table 2: Performance of our models using approaches including Transfer Learning (TL), Data Augmentation (DA), and MultiTask Training (MT).

MultiTask models, we are able to reach better performance on the `oodomain-val` compared to the baseline. We see that after additional out-of-domain fine-tuning on MT model with MLM head trained both `indomain-train` and `ms-marco` (# 10), the QA head is able to achieve the best EM/F1 results on `oodomain-val` compared with other MT models. This demonstrates the potential of the MT approach if we could use sufficiently large and diverse unsupervised training data.

Table 3 shows our final ensemble results that combines the single model predictions on the validation and test leaderboard. Our best model achieved a 40.58/55.68 EM/F1 score on the validation leaderboard, and 45.14/62.16 EM/F1 score on the test leaderboard. It should be noted that while data augmentation and multitask training do not show significant improvement over the baseline by themselves, they do offer additional generalization power when added to the ensemble. On the out-of-domain validation set, we are able to achieve +6.03/+6.26 EM/F1 over the baseline, verifying the effectiveness of our approach.

#	Model Description	val leaderboard		test leaderboard	
		F1	**EM**	**F1**	**EM**
1 EN	1+2+3+4+5	54.16	39.01	61.59	44.34
2 EN	1+2+3+4+5+6+7+11+12+13	**55.68**	**40.58**	**62.16**	**45.14**

Table 3: Performance of our ensemble models on the validation and test leaderboard.

4.4.1 Data Augmentation Ablation Experiment

#	Data Augmentation Techniques	Val F1	Val EM
14 DAA	No Augmentation	52.08	36.65
15 DAA	Synonym Replacement	52.55	36.65
16 DAA	Random Insertion	52.45	36.91
17 DAA	Back Translation	52.55	37.17

Table 4: Ablation experiments using different data augmentation techniques. We report the performance on the `oodomain-val` set.

To analyze different data augmentation techniques, we perform an ablation study by fine-tuning with data augmented using different approaches. We fix the pre-trained model as model #2, and perform additional fine-tuning on the `oodomain-train` data only (#14 DAA), as well as with additional augmented version of `oodomain-train` with synonym replacement (#15 DAA), random insertion (#16 DAA), and back translation (#17 DAA). Other than the training data, all models are trained

with the same hyperparameters with a batch size of 16, learning rate of 3e-05, max length 512 for 10 epochs, validating on `oodomain-val` with `-eval-every` flag set to 100 steps.

We see that in general, training with additional augmented data is able to yield +0.52/+0.47 EM/F1 improvement over training without augmented data. Among the three techniques, adding additional back translation gives the greatest boost in EM. We suspect that this is because we generate paraphrases of the query for back translation while the SR/RI operations are performed on the entire context paragraph. This could cause some answer span being ignored if it is not contained in the augmented context.

5 Analysis

In this section, we demonstrate the qualitative performance of our model in two scenarios: failure cases of our best model and failure cases of baseline model while the best model succeeds. We also include a discussion about the reliability of the dataset ground truth answers and the pitfall of the evaluation metrics.

5.1 Failure Case Analysis

The final ensemble model achieves the best performance in both F1 and EM evaluation metrics; however, it still fails in tasks that require higher level of reading comprehension skills. In the following example, the model fails to give the right answer when it is required to leverage indirect logical associations and prior knowledge related to the question domain that does not appear in the training corpus.

Example 1

Context: The pilot knew that there was nothing he could do to keep the plane long in the air. So he rushed back to where his passengers sat and explained the dangerous situation. In the end he said, "I'm a married man with two small children. I'm sorry to tell you that there are only three parachutes in the plane." And with that he took up one and jumped out. One of the passengers reacted quickly. "I'm a great statesman !" he said. "I've a very bright brain and the world can't do without me!" And with that he jumped out too. The other two passengers, an old man and a young soldier, were quiet for a moment. "Son," the man said, "I'm old and have lived a full life. I'm ready to meet my God." "You'll have to give up that," the young man said, smiling. "The world's smartest man just jumped out with my backpack."

Question: According to the passage, who would be sure to lose his life?
Ground Truth: great statesman
Our Model Output: the young man

There is an indirect logical associations in this example as who would lose his life is not explicitly indicated in the passage. Our model seems to be able to discern that "not having a parachute" would result in death in the given situation as it chooses its answer among the two people who have not gotten the parachutes in the first half of the paragraph. However, the model fails to learn the co-reference between "the smartest man" and the "statesman", and so is unable to figure out the outcome of the statesman.

5.2 Final Model's Improvement over Baseline

Our submitted best model has a substantial improvement over the baseline model in machine reading comprehension problem that requires logical reasoning. In the following example, our best model successfully selects the correct answer from the context but the baseline model does not.

Example 2

Context: South Asia heatwave kills nearly 100 DHAKA - A heat wave sweeping India, Bangladesh and Nepal has killed nearly 100 people over the past two weeks, officials said on June 3, 2005. A third of the people died in northern Bangladesh, mostly women and children from dehydration, heat stroke and diarrhoea. "We are getting reports of several deaths due to heat wave and related diseases almost every day," an official said, as temperatures touched 43℃. The weather office in Dhaka said the hot weather will persist for another week until the monsoon rains which are normally due by the middle of June. Severe heat conditions in the southern Indian have killed at least 55 people, officials in the two states said. While temperatures have fallen from a high of 45℃ in Andhra Pradesh to around 40℃, giving a respite to people, they are still on the rise in Orissa with Talcher town registering 48.5℃, a weather official said. At least five people have died in Nepalfrom extreme heat, the government said.

Question: Which place is the hottest in the early June 2005?
Ground Truth: Talcher
Our Model Output: Talcher town
Baseline Model Output: DHAKA - A heatwave sweeping India, Bangladesh and Nepal

In this example, our model has shown to be able to learn both an association between temperature and location as well as a numerical comparison between temperatures. However, the baseline model selects the first occurrence of a location in the context. This suggests that our model is more capable in extracting and understanding relationships between sentences.

5.3 Pitfalls in Data and Evaluation Methods

In the out-of-domain validation datasets, we observe multiple occurrences of incomplete answers in the ground truth label.

Example 3

Context: If you wish to become a better reader, here are four important things to remember about reading: Knowing why you are reading or what you are reading to find out will often help you to know whether to read rapidly or slowly. Some things should be read slowly throughout. Examples are directions for making or doing something, arithmetic problems, science and history books, which are full of important information. You must read such things slowly to remember each important step and understand each important ideas. Some things should be read rapidly throughout. Examples are simple stories meant for enjoyment, news letters from friends, pieces of news from local, or home-town, papers, telling what is happening to friends and neighbors. In some of your reading, you must change your speed from fast to slow and slow to fast, as you go along. You will need to read certain pages rapidly and then slow down and do more careful reading when you come to important ideas which must be remembered.

Question: Which should be read slowly according to the passage?
Ground Truth: arithmetic problems
Our Model Output: Some things should be read slowly throughout. Examples are directions for making or doing something, arithmetic problems, science and history books

Given the context information, we notice that the ground truth fails to mention `"directions for making or doing something"` and `"science and history books"`. This is an example of non-expert human worker failing to extract comprehensive information from the text. To our surprise, our model trained with data augmentation techniques successfully answers all aspects in the question. Although our model includes an extra sentence, it is able to maintain the completeness of relevant information. This shows that our model has the potential to complement humans for better performance on QA task.

Moreover, we find that our model tends to give more information in the answers than what is required by the question. For example, when the question asks for a city, the ground truth answer would be the city name (e.g. Fremont). However, our model would include the corresponding state (e.g. Fremont, California). In this scenario, the EM metric would label the prediction result as incorrect regardless of the fact that machine's answer is acceptable. Since the number of such examples is non-trivial in the out-of-domain validation dataset, EM might be too strict of a metric for evaluating the QA performance of our model given the ambiguity in ground truth annotations.

6 Conclusion

In this project, we design and implement DAM-Net, a question answering system that has shown robust performance on out-of-domain datasets. DAM-Net combines several approaches, including multi-domain training, additional fine-tuning, multi-task training, and data augmentation through an ensemble. Our experiments have demonstrated the effectiveness of each approach where they offer a substantial improvement in the model's generalization ability to test data beyond the training distribution. We also provide a qualitative analysis on the success and failure cases of our model. Overall, DAM-Net improves the DistilBERT baseline's out-of-domain EM/F1 from **34.55/49.42** to **40.58/55.68** and achieves a top EM/F1 score of **45.14/62.16** on the test leaderboard.

A limitation in our project is that we are not able to incorporate the full `ms-marco` dataset due to hardware constraints. Should there be more RAM available in the future, we would like to investigate the performance impact of a larger training corpus for the MLM auxiliary task. We have also shown that ensembling our models with a simple majority voting scheme is effective in boosting the robustness and prediction accuracy. In the future, we are interested in exploring different ensemble techniques such as weighted voting [29], and stacking [30].

References

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

[2] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding, 2020.

[3] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Hao Tian, Hua Wu, and Haifeng Wang. Ernie 2.0: A continual pre-training framework for language understanding, 2019.

[4] Alon Talmor and Jonathan Berant. Multiqa: An empirical investigation of generalization and transfer in reading comprehension. *CoRR*, abs/1905.13453, 2019.

[5] Yicheng Wang and Mohit Bansal. Robust machine comprehension models via adversarial training. *CoRR*, abs/1804.06473, 2018.

[6] Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. MRQA 2019 shared task: Evaluating generalization in reading comprehension. *CoRR*, abs/1910.09753, 2019.

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

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

[9] Motoki Sato, Hitoshi Manabe, Hiroshi Noji, and Yuji Matsumoto. Adversarial training for cross-domain Universal Dependency parsing. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 71–79, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[10] Shayne Longpre, Yi Lu, Zhucheng Tu, and Chris DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. *CoRR*, abs/1912.02145, 2019.

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

[12] 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.

[13] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension, 2017.

[14] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset. *CoRR*, abs/1611.09830, 2016.

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

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

[17] Jason W. Wei and Kai Zou. EDA: easy data augmentation techniques for boosting performance on text classification tasks. *CoRR*, abs/1901.11196, 2019.

[18] Qizhe Xie, Zihang Dai, Eduard H. Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation. *CoRR*, abs/1904.12848, 2019.

[19] 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, 2020.

[20] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, Aug 1999.

[21] Hongyu Li, Xiyuan Zhang, Yibing Liu, Yiming Zhang, Quan Wang, Xiangyang Zhou, Jing Liu, Hua Wu, and Haifeng Wang. D-net: A pre-training and fine-tuning framework for improving the generalization of machine reading comprehension. pages 212–219, 01 2019.

[22] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension. *CoRR*, abs/1804.09541, 2018.

[23] CS 224N Default Final Project: Building a QA system (Robust QA track), 2021.

[24] Edward Ma. Nlp augmentation. https://github.com/makcedward/nlpaug, 2019.

[25] George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

[26] Tong Niu and Mohit Bansal. Adversarial over-sensitivity and over-stability strategies for dialogue models. *CoRR*, abs/1809.02079, 2018.

[27] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. *CoRR*, abs/1901.11504, 2019.

[28] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. *CoRR*, abs/1611.09268, 2016.

[29] A. Dogan and D. Birant. A weighted majority voting ensemble approach for classification. In *2019 4th International Conference on Computer Science and Engineering (UBMK)*, pages 1–6, 2019.

[30] Mohamed El-Geish. Gestalt: a stacking ensemble for squad2.0, 2020.