Improving Domain Generalization for Question Answering

Stanford CS224N Default Project: Robust QA

Lilia Chang, Jaymee Sheng, Alanna Flores
Department of Computer Science
Stanford University
`{lchang19,jzs,alanna13}@stanford.edu`

Abstract

Domain generalization remains a major challenge for NLP systems. Our goal in this project is to build a question answering system that can adapt to new domains with very few training data from the target domain. We conduct experiments on three different techniques: 1) data augmentation, 2) task-adaptive pretraining (TAPT), and 3) multi-task finetuning. We find that while each individual approach improved out-of-domain question answering compared to the baseline DistilBERT model, their combinations did not necessarily yield additional gains. Our best model implements data augmentation on both in-domain and out-of-domain train datasets with the DistilBERT base model and achieved EM/F1 scores of 35.34/51.58 on the out-of-domain dev set and 42.32/60.17 on the held-out test set. We infer that we've comfortably met our goal of beating the baseline model's performance as the baseline model achieved 32.98/48.14 on the out-of-domain dev set.

1 Introduction

Large-scale pretrained language models have obtained state-of-the-art results on a variety of NLP tasks in recent years. While pretraining on massive unlabeled datasets allows neural models to build general-purpose language representations that are useful for downstream tasks, it is uncertain whether models are learning rules and knowledge that can be generalized to other contexts or simply memorizing the data they were trained on. Reading comprehension is an important task that is at the heart of many practical applications; it is also a comprehensive way of evaluating whether a system truly "understands" a piece of text since many other NLP tasks such as sentiment classification, semantic relation extraction, and information retrieval can be reduced to question answering.

It remains challenging to build a question answering (QA) system that generalizes well to a new target domain without similar training data. Building on previous work in this direction [1, 2, 3], we explore data augmentation, task-adaptive pretraining, and multi-task finetuning as well as the combination of these approaches to improve domain generalization. We implement both word-level augmentation through random inserts and synonym substitution and paragraph-level augmentation through back-translation. We augment 25 percent of our in-domain (ID) train data with word-level augmentation and triple the size of the out-of-domain (OOD) train data through back translation and word-level augmentation. This data augmentation combined with sorting the training data by context length resulted in a 7% performance gain on the OOD dev set.

We also investigate the benefits of additional training using the mask language modeling (MLM) objective, whether during a separate pretraining phase or jointly with question answering during finetuning. We find that either continuing pretraining the baseline model on both ID and OOD train datasets or jointly optimizing the model for MLM and QA during finetuning improves OOD dev performance. However, MLM pretraining followed by multi-task finetuning performed worse than the baseline, possibly due to overfitting on the train data caused by the two-staged MLM training. In the following sections, we provide a brief review of related work (§2), describe our approach (§3) and experiments (§4) in detail, and analyze why certain methods worked well while others did not (§5).

2 Related Work

One way to improve model robustness is through data augmentation. By creating small variances or perturbations in the train data, it encourages the model to learn more generalizable properties instead of brittle correlations. Wei and Zou (2019), for example, use word-level data augmentation techniques including random insertion, swaps, and synonym substitution to improve text classification [4]. Back-translation is another data augmentation technique borrowed from neural machine translation that has seen varying levels of success [1, 5]. In addition to data augmentation, the order of data on which the model is trained also matters. Text examples can vary greatly in difficulty, and training data is usually introduced in a completely random order or grouped by topic such as the SQuAD dataset [6]. Hacohen and Weinshall [7] employ a technique called Curriculum Learning based on the notion that models learn better by seeing easier examples before more difficult ones to improve learning outcomes.

Pretraining general-purpose language representations is increasingly an integral part of building an effective NLP system, from pretrained word embeddings to entire pretrained Transformer-based models like BERT [8]. Recent work has shown that continued pretraining on an unlabeled training set for a given task is an effective approach for enhancing model performance on the target task [2, 9]. A related but different approach is multi-task learning where a system is trained to optimize on multiple tasks simultaneously. Inspired by recent applications of multi-task learning to NLP tasks [3, 10], we take a multi-task approach to finetuning by jointly optimizing the masked language modeling loss and question answering loss using two different architectures.

3 Approach

3.1 Baseline

We finetune the DistilBERT base model [11], a smaller and faster version of the original BERT model [8], for the standard extractive question answering task using all ID training data. This baseline model achieved $EM = 32.98$ and $F1 = 48.14$ on the out of OOD dev set.

3.2 Word-Level Augmentation

We adapt Wei and Zou's implementation to randomly insert and substitute synonyms together. For 15 percent of words in a given context, we replace a word with one of its synonyms and insert a synonym of a random word in the context at a random index. We use the NLTK package to import synonyms from WordNet and perform synonym substitution for non-stop words only. Further, we perform random swaps two times for each context so that some contexts are impacted more than others in contrast to our insertion and substitution technique in which augmentation is proportional to the number of words in the context, in order to make training more difficult.

We tune these three techniques to augment 25 percent of each ID dataset and augment our entire OOD data in addition to including the original OOD data.

3.3 Back-Translation

We also explored back-translation with MarianMT, HuggingFace's multilingual transformer using French and English [12], following the results in Colin Raffel's lecture in which French and English back-translation achieved the highest score [13]. By setting maximum sequence length to 64, we were able to reduce preparation of training data to a reasonable time.

We back-translate every question and answer in the OOD dataset and add this to the training data. Our training data contains triple the amount of the original OOD data: the original OOD sets, the word-level augmented sets and the back-translated sets.

3.4 Example Randomization and Curriculum Learning

We sort the training data more systematically through "curriculum learning" during which training occurs in order of increasing difficulty of the examples. We implement this concept by using the length of contexts as a proxy for their difficulty in the entire training data [14].

3.5 Task-Adaptive Pretraining

We further pretrain the DistilBERT base model on the task-specific training data for question answering. Unlike BERT [8], we omit next sentence prediction and only use the masked LM objective as the latter has been shown to be the more effective one [15]. We use HuggingFace's implementation of the DistilBERT model for masked language modeling. Furthermore, we experiment with masking 15% of random word tokens and replacing them all with the [MASK] token, instead of the 80/10/10 replacement split of [MASK] token, random words, and original words proposed in BERT [8]. All of the pretrained embedding and hidden layers of the Transformer encoder are then passed onto the DistilBERT model for question answering during task-specific finetuning.

3.6 Multi-Task Finetuning

We attempt two different ways of implementing the multi-task finetuning referenced in §2. For each of the models below we refer to the BERT-based MLM and QA models implemented by HuggingFace [11] but implement the combined architectures ourselves in two different ways.

3.6.1 MLM-QA Model

At each forward iteration, we propogate the hidden layer output of the DistilBert model through the MLM and QA model-specific layers and compute the MLM and QA losses. We illustrate the model architecture in Figure 1 (see Appendix). We take the sum of the losses as the total loss. At runtime we mask 15% of the input tokens at random in each batch, and pass through the appropriate labels for computing the MLM loss.

Although this results in the masked inputs also being passed to the QA model, we reason that the QA model should still be able to learn as it sees the unmasked inputs (or differently masked inputs) at some point in time. We recognize, however, that this may mean the QA model learns more slowly or less effectively than if the inputs were unmasked.

3.6.2 MLM-QA, Alternating Model

We address the issue of feeding masked inputs to the QA model by creating a similar model that instead alternates between using MLM and QA layers between batches (MLM-QAlt); see Figure 2 in the Appendix. That is, at each odd-numbered forward iteration, we randomly mask the inputs, feed them through the DistilBert and MLM layers and compute the MLM loss; at each even-numbered forward iteration we do not mask the inputs and feed the inputs through the DistilBert and QA layers and compute the QA loss as normal. In addition to allowing us to feed unmasked inputs to the QA layers this reduces the overall time spent in computing masked inputs. This architecture has the disadvantage of not having the QA layers updated as often.

4 Experiments

4.1 Data

We use three ID reading comprehension datasets, SQuAD [6], Natural Questions from Wikipedia [16], and NewsQA [17] and the three OOD datasets, DuoRC [18], RACE [19], and RelationExtraction [20], for training a question answering system. Our final model will be evaluated on test examples from three different out-of-domain datasets: DuoRC [18], RACE [19], and RelationExtraction [20].

4.2 Evaluation Method

We evaluate our models using exact match (EM) and word-level F1 scores, which are common metrics for extractive question-answering tasks [21]. We choose the model that performs best across the three out-of-domain validation sets. For our data augmentation experiments, we qualitatively analyze the text examples output to inform tuning.

4.3 Experimental Details and Results

We have kept the default hyperparameters with batch size of 16 and learning rate of 3e-05 to measure the effect of implementing the techniques outlined in §3 against the baseline. We train all our models for 3 epochs and compare to the baseline as well as each other.

4.3.1 Data Augmentation

We tried proportional swaps for three and ten percent of words, both of which resulted in decreased F1 and EM scores. We found a slight improvement when augmenting with two swaps per context regardless of context length, likely because too many swaps interrupt answer phrases or create too much noise. We also tested word-level augmentation for the entire ID dataset and included that with the original, doubling our ID data. While that performed well on one epoch, it was too costly to work with and tune given the size of the ID datasets. For this reason, we tried to augment on the word-level for 17 percent of words in a given context for 25 percent of contexts (ID Word-Level Augment in Table 1) and concentrate the remainder of our augmentation on the OOD datasets.

	ID Dev		OD Dev	
Data Augmentation	EM	F1	EM	F1
Baseline	55.12	71.05	32.98	48.14
ID Word-Level Augment	55.07	70.95	33.25	48.43
ID Curriculum Learning	54.57	70.89	34.03	49.06
ID + OOD Back-Translate	54.09	70.51	31.41	48.78
Combination 1	53.95	69.88	31.41	47.56
Combination 2	52.1	68.31	30.37	47.17
Combination 3	53.68	69.41	35.34	51.58

Table 1: Testing Data Augmentation Techniques

In order to increase the difficulty of learning and reduce the possibility of overfitting, we tried shuffling the ordering of the dataset which had little to no effect on our evaluation metrics. This led us to adopt another sorting method, curriculum learning, which proved to be more effective, as shown above.

In addition to word-level augmentation, we first tried back-translation with the `fairseq` transformer [22] and `nlpaug` [23], both of which ran significantly slower than our ultimate choice, HuggingFace's MarianMT model, due to their lack of ability for individual specifications of batch size and maximum sequence length. We tried to augment just a portion of the the ID dataset with back-translation, similar to our word-level augmentation process, but this proved to be too time-consuming. We proceeded to only back-translate the OOD dataset. Augmenting the entire OOD dataset improved the baseline slightly for F1 on the OOD and worsened the outcome for EM (ID + OOD Back-Translate). One possible reason for this is that the OOD datasets were very small in comparison to the ID datasets so their data might just be interpreted as noise and not meaningfully incorporated during training.

Given the success of some of the individual data augmentation approaches, we further experimented with three different combinations. Combination 1 uses the original ID and OOD data, the OOD data with word-level augmentation, and the OOD data back-translated to more meaningfully incorporate the OOD data. This did not work in our favor, possibly because the OOD data was too difficult and noisy in comparison to the ID. We then tried Combination 2, which uses the two augmented versions of the OOD (without the original) plus augmentation on the ID dataset with random swapping on 10 percent of words and insertion and substitution on 15 percent of words in a given context for 33 percent of contexts with the entire training data sorted by context length. This also proved to perform worse possibly because of too much augmentation. In our final combination, we dialed back on ID augmentation, adding insertions/substitutions for 15 percent of the words as well as two random swaps in a given context for 25 percent of contexts, and incorporate the original OOD dataset as well as the two augmented versions. We train on the entire augmented dataset sorted by length and see a huge boost in results and obtained EM/F1 scores of 42.317/60.166 on the test leaderboard.

4.3.2 Task-Adaptive Pretraining

We train the DistilBERT model for masked language modeling on three variants of data: all ID train datasets, a single ID train dataset, and all ID train datasets plus all OOD train datasets. We train each variant for two epochs and use a learning rate of 5e-05 and set maximum sequence length to 256 to speed up training. We observe that training the model for longer leads to overfitting, which makes the model perform worse on the validation data. In addition, we compare our simplified masking approach using 100% [MASK] tokens against the original BERT masking on the combined ID train dataset. As summarized in Table 7 in the appendix, masked LM loss decreased for all domain combinations after further pretraining. We focus on pretraining across all domains because prior work suggests that multi-domain training improves generalizability [1]. Since the original DistilBERT model was not trained on news articles, we also experiment with pretraining only on NewsQA.

	ID Dev		OD Dev	
	EM	F1	EM	F1
Pretrained Model 1	54.21	70.2	31.41	47.92
Pretrained Model 2	54.61	70.23	31.94	46.09
Pretrained Model 3	54.34	70.09	30.1	46.89
Pretrained Model 4	54.9	70.83	34.82	48.7

Table 2: EM and F1 on in-domain and out-of-domain validation for all models that were further pretrained.

We observe that including the OOD training samples during pretraining boosted performance significantly, despite the fact that there were very few OOD training samples (around 1% of the number of ID samples). For models that were pretrained on the combined ID datasets, we see that using the 80/10/10 masking split proposed in BERT [8] (Pretrained Model 3) did worse on out-of-domain validation in terms of EM and only marginally better on F1 compared to the naive masking approach of replacing all masked tokens with [MASK] (Pretrained Model 2). Finally, pretraining on the NewsQA corpus (Pretrained Model 1) did no worse than training on all ID datasets and was six times faster due to the smaller size of the data; in fact, it performed better in terms of F1 on out-of-domain validation. This is not surprising as the baseline model was already trained on Wikipedia texts, the same source that the SQuAD and Natural Questions datasets are drawn from. This suggests that adding training data from a different distribution is more useful for generalizing the QA model to new domains.

4.3.3 Multi-Task Finetuning and Combining Methods

		ID Dev		OOD Dev		OOD Test	
Model	Data Augmentation	EM	F1	EM	F1	EM	F1
Baseline	–	55.12	71.05	32.98	48.14		
Baseline	Combination 3	53.68	69.41	35.34	51.58	42.317	60.166
MLM-QA	–	53.91	69.96	32.72	47.22		
MLM-QAlt	–	53.62	69.87	37.17	51.42	41.491	59.32
MLM-QA	Combination 3	53.73	70.18	31.94	48.41		
MLM-QAlt	Combination 3	53.78	70.41	31.94	47.25	41.881	59.671
MLMPretrain + MLM-QAlt	–	54.52	70.17	29.06	46.55		

Table 3: Metrics from our main models. MLM-QAlt was submitted to the leaderboard under the name `liliaalanaajaymee`; Baseline with Combination 3 data augmentation was submitted to the leaderboard under the name `witt`. The MLM-QAlt ith Combination 3 data augmentation was submitted under the name `combo_test`.

In our experiments we observe that MLM-QAlt performed significantly better than the MLM-QA model. We initially thought that the MLM-QA model would outperform the MLM-QAlt model since it runs through the QA layers at each iteration and thus updates them doubly compared to MLM-QAlt, however the MLM-QA model actually performed worse than the baseline (MLM-QA EM 32.72, FM 47.22 on OOD validation; Baseline EM 32.98, FM 48.14). The MLM-QAlt model in contrast outperformed the baseline significantly (EM 37.17, FM 51.42). We suspect that the MLM-QA model performed worse because we feed masked inputs through the QA layers – we had previously thought that masking inputs through QA would not be consequential, and might actually benefit performance since it could be like another form of drop-out. The masking of inputs seems however to be the most likely explanation for why MLM-QA could not perform as well as the baseline model.

As stated in §3.6.2, MLM-QAlt has the advantage over MLM-QA of not having to mask inputs on every run. Because of this, one might suspect that it would take MLM-QAlt twice the number of iterations as the baseline to converge. This appears to not be the case: MLM-QAlt was able to learn as quickly as the baseline model in terms of negative log-likelihood convergence. Jointly optimizing with the MLM task seems to complement the learning process for the QA task.

Finally, we combine the MLM-QAlt architecture with our best-performing data augmentation methods (Combination 3) and best-performing MLM pretrained model (Pretrained Model 4), respectively. We keep the same hyperparameters in training so that we can isolate the effect of combining the systems. Both combinations performed worse than each standalone model on OOD dev. For MLMPretrain + MLM-QAlt, it is possible that additional training using masked language modeling during the multi-task finetuning stage led to overfitting; reduced training time for the QA task may also have contributed to the underperformance of MLM pretraining followed by MLM-QAlt. Data augmentation combined with MLM-QAlt also did not lead to further performance gains. We explore the reasons why in more detail in the next section.

5 Analysis

Baseline predictions	Data Augmentation predictions
Yeong-mi (Doona Bae)	Yeong-mi
Franken and Van Gein	Van Gein
Seiji Hasumi	Hasumi
Deepika Padukone)	Deepika Padukone
Varano de' Melegari, near Parma, Italy	Varano de' Melegari
Santa Rosa, California	Santa Rosa

Table 4: Examples where data augmentation is able to find an exact match to the answer while the baselineis unable to.

We first consider why the augmented ID and OOD datasets perform so well against the baseline. Many of the predictions resemble those in Table 4 where the augmented version makes the correct prediction and the baseline does not. It seems that the augmented version is much better at parsing out the excess information, likely due to the increase of difficulty during training and also exposure to many possible forms in which out of domain answers can be presented, including synonyms, paraphrased versions from back-translation, and the original.

We can find evidence to support our hypothesis about why MLM-QAlt outperforms MLM-QA upon inspecting the OOD validation predictions. In Table 5, we list examples where MLM-QAlt retreieved the exact answer while MLM-QA could not. In these examples, MLM-QAlt selects shorter phrases as the answer (examples 2 and 3) and is better at identifying the exact part of the phrase from which the answer should be derived (examples 1 and 3). Perhaps MLM-QA selects longer phrases because it can increase its recall by being more liberal with its predicted answer lengths in the absence of masked words.

While we expected that combining the best performing data augmentation combination (Combination 3) and MLM-QAlt architecture would yield even better results, the combined model did not do significantly better on the OOD validation set and did comparably well to the underlying models on the OOD test set. This system seems to output the first thing that could feasibly be the right answer, normally if a key word in the question appears in the answer, without really understanding the context or considering that the correct answer may be later in the context. We include examples of this which occurs nearly whenever the best data augmentation model and the MLM-QA-Alt predict the exact correct answer, but the combined model does not (see Table 6).

Question	Context snippet	MLM-QA pred.	MLM-QAlt pred.
Which kind of the following persons will be the first to be employed if computers continue to develop?	...Computers in United States have already begun to take the place of workers whose tasks are simple. The variety of jobs, done only by humans in the past... increases ... We will be faced with mass unemployment for all but a handful of highly trained professionals....	workers whose tasks are simple. The variety of jobs, done only by humans	highly trained professionals
Which team is Thatcher Szalay a member of?	Thatcher Szalay (born January 18, 1979) is an American football player of Hungarian origin who has previously played for the NFL on the Baltimore Ravens, Seattle Seahawks, and the Bengals.	NFL on the Baltimore Ravens	Baltimore Ravens
What's this passage about?	The French Revolution broke out in 1789....On July 14, 1789, they stormed and took the Bastille, where political prisoners were kept. Ever since that day, July 14 has been the French National Day. ...	French National Day	French Revolution

Table 5: Examples of question, context, predict answer tuples from the OOD validation set where MLM-QAlt retrieved the exact answer while MLM-QA could not.

Question	Context snippet	Answer	MLM-QAlt with data aug. pred.
Who might block the development of Brazil?	Brazil already had the potential to develop. The Brazilian Empire, Pedro I, abolished slavery … Though Brazil always tried to maintain democracy, it was failed several times by the dictatorship of Getulo Vargas.	Getulo Vargas	Brazil already had the potential to develop. The Brazilian Empire
Which subjects was the writer poor at?	I was to ... teach them all subjects–including art, football, cricket and so on–in turn at three different levels. Actually, I was depressed at the thought of teaching algebra and geometry–two subjects in which I had been rather weak at school.	algebra and geometry	art, football, cricket and so on
What is the name of the chromosome where you can find NKG2D?	NKG2D is encoded by KLRK1 gene which is located in the NK-gene complex (NKC) situated on chromosome 6 in mice and chromosome 12 in humans.	chromosome 12	chromosome 6

Table 6: Examples of question, context, answer tuples from the OOD validation set where the combined model (MLM-QAlt with augmentation) was incorrect. In each of these it seems that the combined system identified just the first phrase or clause that could possibly be correct.

The combined model at times predicts the correct answer when other models do not. This, on average, occurs for shorter contexts. The average length of contexts where the combined model outperforms the others on the validation set is 1287.25; the average length of contexts in the validation set overall is 1781.58; the average length of contexts where the underlying systems are correct as opposed to the combined model is 2175.13. It is likely that this model performs better on shorter contexts because there is less likely to be an earlier sentence discussing similar words to those in the question that the model could mistake as the right answer.

We suspect that the combined system was unable to learn to generalize to OOD because having the augmented data created too much noise, or created too much spurious data for the system to parse apart, given that the QA layers are passed through for fewer number of iterations than the original architecture. The MLM-QAlt model with the original input data seemed to be able to efficiently learn from the available data, and so perhaps using the various augmentation techniques resulted in too much noise to contribute to meaningful learning.

6 Conclusion

We tried many approaches in tackling the problem of producing a QA system that is robust to OOD samples. We found that simply augmenting the ID and OOD training samples available to us, specifically using insertions, substitutions, swaps and back-translations, boosted our model performance with just the baseline model architecture significantly. Further pretraining using the masked LM objective on the few OOD training samples also proved to be helpful for improving generalization. We also explored various model architectures in the realm of multi-task learning and found that jointly optimizing the QA loss with MLM loss allowed the model to generalize on the OOD samples significantly, confirming existing literature surrounding multi-task learning [3]. Hoping that these gains from data augmentation, adaptive pretraining, and multi-task learning would be additive, we tried combining the techniques but found that the sum of the techniques performed only slightly better and sometimes worse than the smaller underlying systems alone.

There are many avenues for future work. For data augmentation, we currently sort by the length of contexts as a proxy for the difficulty of text samples, but future work could consider more nuanced measures of the difficulty of a textual example such as the usage of rare words and the scale of learning target [14]. Additionally, instead of generating three times the original OOD data through augmentation techniques, we could consider augmenting OOD datasets based on their proportion in the test set, to put greater weight on domains in which it may be more important to generalize well to. Finally, instead of randomly augmenting a portion of the ID contexts, we could attempt to augment only easier contexts or only more difficult contexts, as measured by the aforementioned metrics, to see if this improves overall training.

The number of experiments we were able to conduct with pretraining was limited given its time-consuming nature. For future work, one may consider pretraining on the augmented OOD datasets, alone or combined with ID data, or explore other language modeling objectives that may be more beneficial for the downstream question answering ask. Future work could also be devoted to trying various hyperparameters generally, but especially for the combined MLM-QAlt with data augmentation system. If our hypothesis of MLM-QAlt not being able to handle the amount of noise generated by augmented data is correct, we could perhaps try various hyperparameters to produce a more suitable amount of augmented data for this model.

References

[1] Shayne Longpre, Yi Lu, Zhucheng Tu, and Chris DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering, 2019.

[2] 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.

[3] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy, July 2019. Association for Computational Linguistics.

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

[5] 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, 2018.

[6] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics.

[7] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. *CoRR*, abs/1904.03626, 2019.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[9] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[10] 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. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 212–219, Hong Kong, China, November 2019. Association for Computational Linguistics.

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

[12] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.

[13] Colin Raffel. T5 and large language models: The good, the bad, and the ugly.

[14] Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online, July 2020. Association for Computational Linguistics.

[15] Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[16] 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.

[17] Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, August 2017. Association for Computational Linguistics.

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

[19] Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[20] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[21] Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. Mrqa 2019 shared task: Evaluating generalization in reading comprehension, 2019.

[22] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

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

Appendix

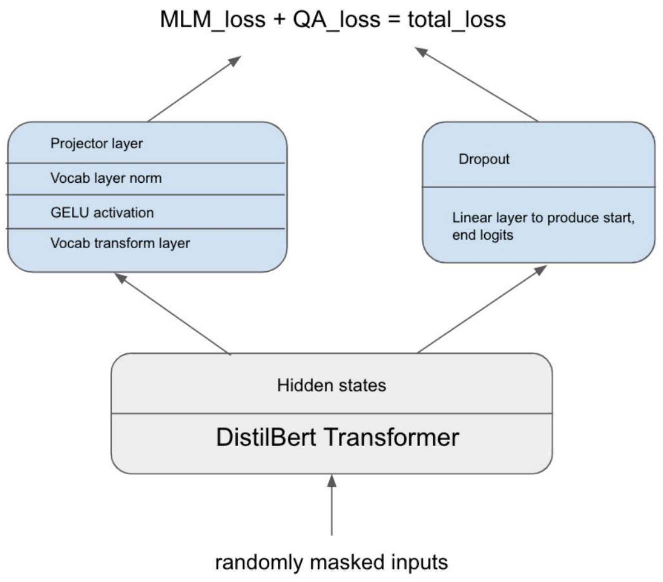

Figure 1: Model architecture of the MLM-QA model. The MLM and QA layers each share the hidden states as input from the DistilBert base model. We then compute the loss as the sum of the respective model losses. Recall that the inputs are randomly masked before going through the DistilBert base.

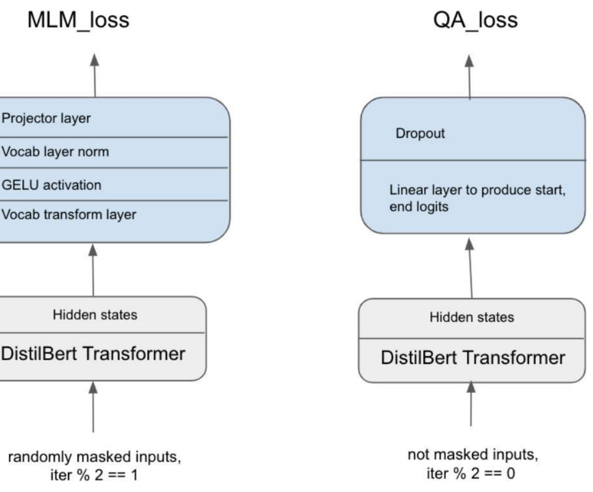

Figure 2: Model architecture of the MLM-QAlt model.

	Domain	Mask	$\mathcal{L}_{DistilBERT}$	$\mathcal{L}_{TAPT}$
Pretrained Model 1	NewsQA	100% [MASK]	2.53	1.99
Pretrained Model 2	SQuAD, NewsQA, Natural Questions	100% [MASK]	2.15	1.49
Pretrained Model 3	SQuAD, NewsQA, Natural Questions	80/10/10	2.15	1.47
Pretrained Model 4	SQuAD, NewsQA, Natural Questions, DuoRC, RACE, RelationExtraction	100% [MASK]	2.36	1.5

Table 7: Masked LM loss on the validation set for the DistilBERT model before ($\mathcal{L}_{DistilBERT}$) and after ($\mathcal{L}_{TAPT}$) additional pretraining.