PPDAC: A Plug-and-Play Data Augmentation Component for Few-shot Extractive Question Answering

Qi Huang[1,2] **Han Fu**[2] **Wenbin Luo**[1] **Mingwen Wang**[1,2] **Kaiwei Luo**[2]
[1] School of Computer and Information Engineering, Jiangxi Normal University,
Nanchang, Jiangxi 330022, China
[2]School of Digital Industry, Jiangxi Normal University,
Shangrao, Jiangxi 334000, China
{huangqi, 202141600077, lwb, mwwang, luokaiwei}@jxnu.edu.cn

Abstract

Extractive Question Answering (EQA) in the few-shot learning scenario is one of the most challenging tasks of Machine Reading Comprehension (MRC). Some previous works employ external knowledge for data augmentation to improve the performance of few-shot extractive question answering. However, there are not always available external knowledge or language- and domain-specific NLP tools to deal with external knowledge such as part-of-speech taggers, syntactic parsers, and named-entity recognizers. In this paper, we present a novel Plug-and-Play Data Augmentation Component (PPDAC) for the few-shot extractive question answering, which includes a paraphrase generator and a paraphrase selector. Specifically, we generate multiple paraphrases of the question in the (question, passage, answer) triples using the paraphrase generator and then obtain highly similar statements via paraphrase selector to form more training data for fine-tuning. Extensive experiments on multiple EQA datasets show that our proposed plug-and-play data augmentation component significantly improves question-answering performance, and consistently outperforms state-of-the-art approaches in few-shot settings by a large margin.

1 Introduction

Extractive Question Answering (EQA) presents a significant challenge within the domain of machine reading comprehension, requiring a nuanced understanding of both questions and passages to accurately identify the text fragments that serve as answers. High-performance EQA algorithms are essential for enabling retrieval systems to swiftly locate relevant passages and text fragments based on user queries. However, the reality is that not all scenarios offer a wealth of labeled samples, leading to diminished effectiveness of current EQA algorithms. This gap underscores the importance of investigating EQA algorithms within few-shot learning contexts, which are not only more aligned with real-world situations but also more feasible for practical implementation.

The primary issue confronting existing EQA algorithms in few-shot scenarios is performance degradation due to a disconnect between the pre-training objectives of Masked Language Modeling (MLM) and the fine-tuning objectives of downstream Machine Reading Comprehension (MRC) tasks. While (Glass et al., 2020; Ram et al., 2021; Chada and Natarajan, 2021) have attempted to bridge this gap by integrating downstream tasks into pre-training or by aligning the fine-tuning framework with the pre-training framework (known as the prompt-based fine-tuning paradigm, or prompt-tuning), these approaches often require extensive computational resources and overlook the potential of external knowledge bases for data augmentation in few-shot EQA.

In response to these challenges, another avenue of research has focused on leveraging external knowledge to either generate training data or enhance the representation of prompt-tuning paradigm methods. Specifically, (Lewis et al., 2019) generate context, question, and answer triples via the traditional NLP pipeline from the external knowledge base, which selects noun phrases and named entities in Wikipedia paragraphs as potential answers, and then masks from the context to create pseudo-questions. Moreover, (Wang et al., 2022) proposed a novel framework named knowledge enhanced contrastive prompt-tuning, which enhances the representation of the question by the support of the rich semantics from the external knowledge base and the passage context. Despite the potential of these approaches, they face significant hurdles, including limited availability of external knowledge or language- and domain-specific NLP tools, the risk of error accumulation in the NLP pipeline, and a scarcity of data augmentation methods specifically designed for few-shot EQA tasks.

To address these issues, this paper introduces a novel Plug-and-Play Data Augmentation Component (PPDAC), aimed at generating paraphrases of questions within (question, passage, answer) triples to enrich the training data. The PPDAC comprises a paraphrase generator, a paraphrasing filter, and a paraphrase selector, which work together to transform the original question statements into multiple new statements with similar semantics but varied syntactic structures. This approach substantially increases the volume of training data available for fine-tuning. Our experimental evaluation across various EQA baselines in few-shot scenarios reveals significant performance improvements, showcasing the effectiveness of PPDAC in addressing the challenges of few-shot EQA without relying on external knowledge bases.

In summary, our contributions can be summarized as follows:

- We present a novel Plug-and-Play Data Augmentation Component (PPDAC) for data augmentation in few-shot EQA tasks.
- PPDAC replaces the external knowledge base by converting the statements of the questions in the data (questions, paragraphs, answers) into multiple new statements and selecting highly similar statements to form more training data for fine-tuning, thereby improving the applicability of data enhancement methods.
- Experimental results show that our proposed PPDAC can effectively improve the performance of few-shot EQA algorithms.

2 Related Work

Extractive Question Answering (EQA) methods based on pre-trained language models resulted in performance close to (and sometimes exceeding) a human performance when fine-tuned on several QA benchmarks(Kenton and Toutanova, 2019; Brown et al., 2020; Bao et al., 2020; Raffel et al., 2020). These methods need to be fine-tuned on tens of thousands of examples to obtain this result. However, Only a small amount of annotated training data is available in a more realistic and practical scenario, which results in their performance degrading significantly. To address this issue, researchers started from two aspects: 1) reducing the difference between the pre-training objective of Masked Language Modeling (MLM) and the fine-tuning objective of downstream MRC tasks; 2) utilizing external knowledge to generate the training data or enhancing the representation.

To reduce the difference between the pre-training objective and the fine-tuning objective, some researchers construct new pre-training tasks specifically tailored to the EQA task, and other researchers propose a new fine-tuning framework that is directly aligned with the pre-training framework. For instance, (Ram et al., 2021) designed a new pretraining scheme tailored for few-shot question answering, i.e. using a recurring span selection as the objective of pre-training models. (Castel et al., 2021) presents a decoding algorithm on the fine-tuning phase that efficiently finds the most probable answer span in the passage. And (Chada and Natarajan, 2021) explore a simple fine-tuning framework that leverages pre-trained text-to-text models and is directly aligned with the pre-training framework, in terms of both the input-output design and the training objective. However, these methods may cost a lot of computational resources during the pre-training phase and ignore the external knowledge base for data augmentation in few-shot question answering.

Therefore, some researchers explore another branch of the few-shot EQA that studies the data augmentation methods to generate the training data or enhance the representation from the external knowledge base. (Lewis et al., 2019; Glass et al., 2020) exploit the traditional NLP pipeline to select noun phrases and named entities in Wikipedia paragraphs as potential answers, and then mask these potential answers from the context to create pseudo-questions. The difference is that (Lewis et al., 2019) utilizes unsupervised machine translation methods to translate these pseudo-questions into real ones, while (Glass et al., 2020) exploits information retrieval to find new text passages that can answer the pseudo-questions. Moreover, (Wang et al., 2022) presents a novel framework named knowledge enhanced contrastive prompt-tuning, which enhances the representation of the question by the support of the rich semantics from the external knowledge base and the passage context. However, the above studies assume access to external knowledge or language- and domain-specific NLP tools such as part-of-speech taggers, syntactic parsers, and named-entity recognizers, which may not always be available.

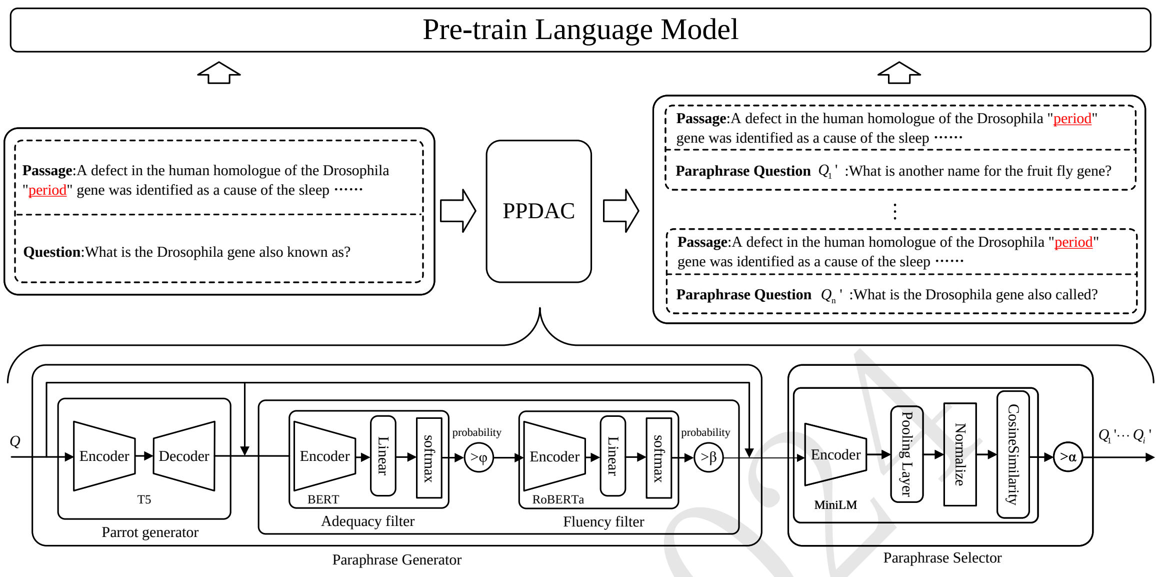
Pre-train Language Model
Passage:A defect in the human homologue of the Drosophila "period" gene was identified as a cause of the sleep ……
Question:What is the Drosophila gene also known as?
PPDAC
Passage:A defect in the human homologue of the Drosophila "period" gene was identified as a cause of the sleep ……
Paraphrase Question Q1' :What is another name for the fruit fly gene?
Passage:A defect in the human homologue of the Drosophila "period" gene was identified as a cause of the sleep ……
Paraphrase Question Qn' :What is the Drosophila gene also called?
Q
Encoder
Decoder
T5
Parrot generator
Encoder
Linear
softmax
BERT
probability
>φ
Adequacy filter
Encoder
Linear
softmax
RoBERTa
probability
>β
Fluency filter
Paraphrase Generator
Encoder
MiniLM
Pooling Layer
Normalize
CosineSimilarity
>α
Q1'··· Qi'
Paraphrase Selector

Figure 1: The overview of the PLM-based framework for few-shot EQA with our proposed component PPDAC. Given a (question, passage, answer) triples, our proposed PPDAC converts the statement of the question in (question, passage, answer) triples into multiple new statements by the paraphrase generator, and then obtains highly similar statements via paraphrase selector to form more training data for fine-tuning the pre-train language model.

3 The Proposed Plug-and-play Component

In this section, we formally introduce the few-shot EQA task and our proposed plug-and-play component in detail. The overview of the PLM-based framework for few-shot EQA with our proposed component is shown in Figure 1.

3.1 Task Statement

Given a passage $P = (p_1, \ldots, p_n)$ and the corresponding question $Q = (q_1, \ldots, q_m)$, the goal of EQA is to find a sub-string of the passage as the answer $A = (p_k, \ldots, p_l)$, where n and m represent the lengths of the passage and the question, respectively. Here, $p_i(i = 1, \ldots, n)$ and $q_j(j = 1, \ldots, m)$ denote the tokens in P and Q, respectively. k and l refer to the start and end position of the answer, restricted to $[1, n]$. The conventional EQA methods based on language models need to be finetuned on tens of thousands of samples to achieve satisfactory performance. However, only a few samples (P, Q, A) can fine-tune the pre-trained language models in the few-shot scenarios, i,e. few-shot EQA task. Rather than the methods exploiting external knowledge for few-shot EQA, we explore a plug-and-play data augmentation component to generate training data for fine-tuning the pre-trained language models. In the following, we will introduce the details of our proposed plug-and-play data augmentation component, which consists of a paraphrase generator and a paraphrase selector. Subsequently, we provide a comprehensive exposition of these individual components.

3.2 Paraphrase Generator

The Paraphrase Generator is the cornerstone of our proposed component inspired by (Damodaran, 2021), which aims to convert the question of (question, passage, answer) triples into a series of new statements and preserve the core meaning of the question. This generator encompasses three essential components: parrot generator, adequacy filter, and fluency filter. The parrot generator utilizes a T5 model with a beam search strategy to generate various statements of question. Then the adequacy filter calculates the adequacy score between the generated statements and the original question via a pre-trained model to filter out the statements below the threshold. Finally, the fluency filter exploits a pre-trained model to evaluate the fluency score of the statements and filter out the statements below the threshold.

3.2.1 Parrot Generator

To generate the paraphrase of the question in (question, passage, answer) triples, the parrot generator utilizes a transformer-based T5 model for the task of autoregressive language generation. the parrot generator exploits the encoder of T5 model to transform the given question Q into a representation X containing contextual semantic information. The formalization process is as follows:

$$X_Q = Encoder_{T5}(Q) \tag{1}$$

Next, according to the contextual semantic representation X_Q, the parrot generator uses the decoder of the T5 model to decode the semantic representation to obtain a paraphrase of the original question. To enhance the diversity of the generated paraphrase, we introduce the beam search algorithm to decoding. Specifically, during the decoding process, the beam search algorithm preserves the num beam most likely probable, that is a beam represents a word, in each time step. Then when the search ends, the beam sequence or say the word sequence with the highest cumulative probability is selected as the output of the algorithm. The above process increases the likelihood of generating sequences of high-probability word combinations. However, the sentence generated by the beam search algorithm still contains repetitive sequences of words.

To this end, inspired by the n-grams penalty introduced in (Paulus et al., 2018; Klein et al., 2017), we utilize a diversity penalty parameter to control the recurrence of words. The diversity penalty parameter is determined by comparing the currently generated token with a beam from another group at a specific time. If there is a match between the generated token and the prior beam then remove the token from the current beam. In summary, the above operations ensure that the parrot generator can generate sentences that are fluent and have as few repeated words as possible.

3.2.2 Paraphrase Filter

In our approach, we enhance the paraphrase generation process by incorporating two critical filters: an adequacy filter and a fluency filter. Both filters leverage finely-tuned pre-trained models to ensure the high quality of generated paraphrases, aligning them closely with the semantic intent and linguistic standards of the original questions.

Adequacy Filter The adequacy filter employs the parrot_adequacy_model, a specialized textual entailment classifier derived from fine-tuning the pre-trained RoBERTa model. This model adeptly identifies whether the generated paraphrase maintains the same semantic essence as the original question, categorizing relationships as entailment, contradiction, or neutral. We utilize this classifier to assess the neutrality of paraphrases, discarding those that do not meet a predefined neutrality threshold. This ensures that only semantically consistent paraphrases are retained [0].

Fluency Filter Simultaneously, the fluency filter applies the parrot_fluency_model, a binary classifier fine-tuned from the pre-trained BERT model, designed to evaluate the linguistic fluency of paraphrases. By quantifying the fluency score, this model filters out paraphrases that fall below a certain fluency threshold, ensuring that the final selections are not only semantically accurate but also linguistically polished [1].

[0]The $parrot\_adequacy\_model$ is available for community use at https://huggingface.co/prithivida/parrot_adequacy_model.

[1]The $parrot\_fluency\_model$ is available for community use at https://huggingface.co/prithivida/parrot_fluency_model.

By integrating these filters, our method significantly enhances the quality of paraphrases used in training, directly contributing to the improved performance of our few-shot EQA system. The open-source availability of both models underscores our commitment to transparency and community collaboration.

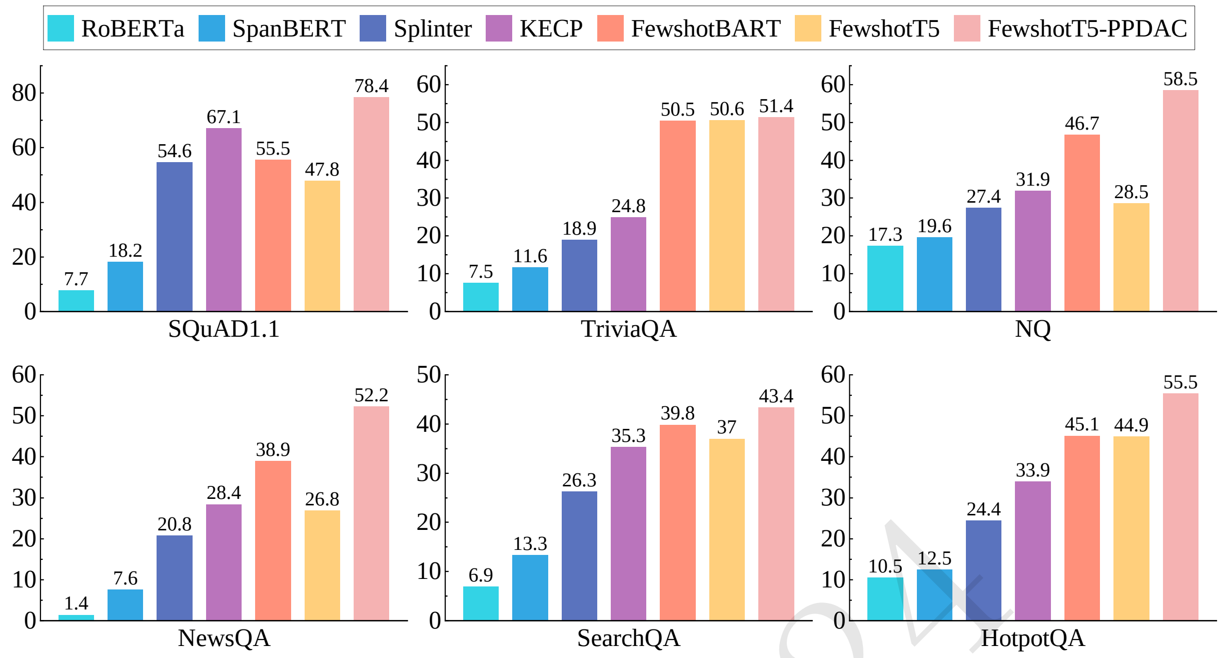

Figure 2: The performance of different methods over 6 datasets in the 16-shot setting, where FewshotT5-PPDAC is the FewshotT5-based model with our PPDAC. The value of performance is the mean of the F1 value across 5 randomly sampled training sets of the same size.

3.3 Paraphrase Selector

To obtain a high-quality paraphrase of the question for fine-tuning the pre-trained language model, we designed a paraphrase selector to remove redundancy and enhance the diversity of the question statement. The paraphrase selector assesses the diversity and quality of generated questions by calculating the similarity between the generated questions and the original questions. The encoding of both the generated and original questions is conducted utilizing the pre-trained MiniLM. The formula of similarity is as follows:

$$X_Q = Encoder_{MiniLM}(Q) \tag{2}$$

$$X_{Q_i'} = Encoder_{MiniLM}(Q_i') \tag{3}$$

$$\cos_{Q,Q_i'} = \frac{X_Q \bullet X_{Q_i'}}{\sqrt{{X_Q}^2} \times \sqrt{X_{Q_i'}^2}} \tag{4}$$

where Q_i' represents the i th question generated by the paraphrase generator, i=[1,...,n]. Then we select the generate questions Q' whose value of $cos_{Q,Q_i'}$ is greater than α to form the (Q', P, A) for fine-tuning, α is our defined threshold of similarity.

As we all know, the value of the α is the key factor to maintaining a dynamic balance between the diversity and quality of new questions generated. Therefore the selection of α value is essential for our proposed component.

Although both the Paraphrase Filter and the Paraphrase Selector are involved in filtering high-quality questions, they serve different yet complementary roles within the augmentation pipeline. The Paraphrase Filter acts as the first gatekeeper, ensuring semantic consistency and fluency. The Paraphrase

	Dataset	Train	Dev	Test
split I	SQuAD	86,588	10,507	-
	NewsQA	74,160	4,212	-
	TriviaQA	61,688	7,785	-
	SearchQA	117,384	16,980	-
	HotpotQA	72,928	5,904	-
	NQ	104,071	12,836	-
split II	BioASQ	-	1,504	1,518
	TextbookQA	-	1,503	1,508

Notes: the test data of the Split II in the MRQA 2019 shared task is not available.

Table 1: The statistics of the two subsets of the MRQA 2019 shared task.

Selector further optimizes the output by enhancing the overall quality of the questions. This division of responsibilities prevents redundancy and ensures that the generated questions are both accurate and natural, thereby improving the quality of the augmented data.

4 Experiments

In this section, we conduct extensive experiments to verify the improvement of the proposed plug-and-play component PPDAC for the few-shot EQA methods[2].

4.1 Baselines

To thoroughly evaluate the enhancement brought by our proposed component to few-shot EQA methods, we benchmark against a broad spectrum of methods, including those introduced beyond 2019 to ensure our comparison encapsulates the latest advancements in the field:

1) **RoBERTa** (Liu et al., 2019): A pre-trained model that refines BERT by implementing a dynamic masking strategy, establishing a strong baseline for numerous NLP tasks.

2) **SpanBERT** (Joshi et al., 2020): A variant of BERT that focuses on predicting spans of text, rather than individual tokens, for improved performance on span-based tasks like EQA.

3) **Splinter** (Ram et al., 2021): Pioneers the use of recurring span selection as a pre-training task specifically tailored for enhancing EQA performance.

4) **FewshotBART** and **FewshotT5** (Chada and Natarajan, 2021; Raffel et al., 2020; Lewis et al., 2020): Leverage the respective strengths of BART and T5 within an aligned fine-tuning framework, showcasing their adaptability to few-shot learning contexts.

5) **FewshotBARTL** (Chada and Natarajan, 2021; Lewis et al., 2020): Within the identical fine-tuning framework, the BART-large model possesses a parameter count significantly exceeding that of both BART-base and T5-base models. Consequently, its outcomes are utilized merely for comparative reference.

6) **KECP** (Wang et al., 2022): A method that transforms the EQA task into a non-autoregressive Masked Language Modeling generation problem, enhances the representation of the question by the external knowledge base and trains the model using the MLM and contrastive learning objectives.

4.2 Datasets

We evaluate our proposed component on a subset of the MRQA 2019 shared task (Fisch et al., 2019), which contains extractive question-answering datasets in a unified format and the answer is a single span in the given text passage. The MRQA 2019 shared task is split into two subsets (i.e. Split I and Split II) and the statistics of each dataset on two subsets are shown in Table 1.

Split I of the MRQA 2019 shared task: It is a shared task consisting of 6 EQA datasets: SQuAD1.1 (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2017), TriviaQA (Joshi et al., 2017), SearchQA (Dunn et al., 2017), HotpotQA (Yang et al., 2018), and Natural Question (Kwiatkowski et al., 2019). Following previous works (Ram et al., 2021; Wang et al., 2022), we utilize the official development set for evaluation, as the testing set is not publicly available.

[2]The source codes and datasets used will be available on the GitHub repository after being accepted.

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16 Examples								
Splinter	54.6±6.4	18.9±4.1	27.4±4.6	20.8±2.7	26.3±3.9	24.0±5.0	28.2±4.9	19.4±4.6
Splinter-PPDAC	**58.2±2.5**	**21.4±4.8**	26.9±2.2	19.3±2.2	**28.7±2.1**	**28.8±2.6**	**28.8±1.0**	19.4±3.9
FewshotBART	55.5±2.0	50.5±1.0	46.7±2.3	38.9±0.7	39.8±0.1	45.1±1.5	49.4±0.1	19.9±1.2
FewshotBART-PPDAC	**57.4±0.8**	47.1±0.1	**47.7±0.3**	37.8±0.6	**45.9±1.2**	**60.4±1.1**	**49.9±1.8**	**20.1±1.0**
FewshotT5	47.8±6.9	50.6±4.9	28.5±14.5	26.8±2.7	37.0±3.3	44.9±3.5	46.3±5.9	25.9±5.0
FewshotT5-PPDAC	**78.4±0.2**	**51.4±2.0**	**58.5±1.0**	**52.2±0.5**	**43.4±2.1**	**55.5±0.3**	**58.6±0.4**	**32.1±2.5**
128 Examples								
Splinter	72.7±1.0	44.7±3.9	46.3±0.8	43.5±1.3	47.2±3.5	54.7±1.4	63.2±4.1	42.6±2.5
Splinter-PPDAC	**73.8±0.7**	**45.1±1.0**	**48.8±0.4**	**43.6±0.9**	**48.0±2.1**	**55.3±0.5**	**64.4±2.0**	37.9±2.0
FewshotBART	68.0±0.3	50.1±1.8	53.9±0.9	47.9±1.2	58.1±1.4	54.8±0.8	68.5±1.0	29.7±2.4
FewshotBART-PPDAC	**71.6±1.0**	**53.9±1.2**	**54.1±0.2**	**49.9±0.1**	**59.1±0.2**	**67.1±0.6**	**72.4±0.2**	**37.8±0.4**
FewshotT5	64.6±6.1	51.7±3.1	47.0±4.6	40.0±1.9	57.0±4.5	56.1±3.7	68.2±3.6	33.6±2.1
FewshotT5-PPDAC	**86.3±0.1**	**52.8±0.3**	**64.3±0.1**	**60.9±0.4**	**60.8±0.4**	**64.4±0.9**	**73.9±0.9**	**51.5±0.8**
1024 Examples								
Splinter	82.8±0.8	64.8±0.9	65.5±0.5	57.3±0.8	67.3±1.3	70.3±0.8	91.0±1.0	54.5±1.5
Splinter-PPDAC	82.8±0.7	**64.9±1.0**	**66.2±0.4**	57.0±0.9	67.3±1.1	**70.4±0.5**	91.0±0.5	**54.7±1.3**
FewshotBART	79.6±0.1	55.5±0.7	61.4±0.1	58.9±0.6	70.0±0.1	65.8±0.7	92.2±0.8	50.0±0.2
FewshotBART-PPDAC	**79.7±0.1**	55.5±0.9	**62.4±0.1**	58.7±0.1	**70.5±0.2**	65.0±0.6	**92.5±0.2**	**51.8±0.4**
FewshotT5	88.6±0.2	60.0±0.5	69.4±0.3	64.7±0.2	73.0±0.2	70.2±0.2	86.9±0.2	59.7±0.5
FewshotT5-PPDAC	**89.0±0.1**	**61.0±0.1**	69.4±0.1	**64.9±0.1**	**73.1±0.1**	**71.2±0.2**	**87.8±0.1**	**60.1±0.6**

Notes: this is a part of the whole experimental results, the whole please consult the Appendix

Table 2: The gain results of the PPDAC components are compared with the baseline results -PPDAC indicates the baseline with our PPDAC. The standard deviation is calculated across 5 runs with different seeds. NQ stands for Natural Questions dataset. The table highlights improvements in effectiveness with bolded numbers, while the best results achieved with the same dataset and number of training samples are denoted in blue.

Split II of the MRQA 2019 share task: Following (Ram et al., 2021), we also select two datasets from Split II of the MRQA 2019 shared task that were annotated by domain experts to evaluate our proposed component: namely BioASQ (Tsatsaronis et al., 2015) and TextbookQA (Kembhavi et al., 2017). These datasets only included publicly available development sets within the MRQA platform, each containing approximately 1,500 examples. For evaluation, we sample 400 examples from each dataset to create a separate test set, and the remaining data serves as the training data.

4.3 Implementation Details

Our experimental framework adopts a methodology similar to (Ram et al., 2021) for constructing few-shot training and development sets across various EQA datasets. We achieve this by randomly selecting K samples from the original training sets, where K ranges from 16 to 1024. Due to the unavailability of the original test sets, evaluations are conducted on the complete development sets.

Baseline models are selected based on pre-trained language models (PLMs) available through HuggingFace, with initial hyperparameters set accordingly. Model training employs the Adam optimizer(Kingma, 2014) with bias-corrected moment estimates, as recommended for few-shot learning contexts(Zhang et al., 2021). The training learning rate is fixed at 2e-5, incorporating a warm-up phase covering the initial 10% of steps, succeeded by a linear decay. Training durations are set to 40 epochs or 200 steps, whichever is greater, for experiments involving up to 1024 examples. The batch size is configured at 8 for training with 128 or fewer examples and reduced to 4 for 1024-shot learning scenarios.

For the paraphrase selector, the threshold $\alpha$is set to 0.9 based on the experience of a large number of experiments. To ensure the robustness of our findings, experiments are conducted with five distinct random seeds $\{42, 43, 44, 45, 46\}$, and performance metrics are averaged. The evaluation metric employed is the F1 score, measuring the average overlap between predicted and true answer texts, following the protocol established by (Ram et al., 2021). Our models are implemented using PyTorch and trained on an RTX 3090 GPU server, ensuring computational efficiency and reproducibility of results.

4.4 Few-shot EQA performance

Our comprehensive evaluation showcases the significant enhancements brought about by the Plug-and-Play Data Augmentation Component (PPDAC) in few-shot Extractive Question Answering (EQA) settings. Figure 2 vividly illustrates the superior performance of FewshotT5-PPDAC, our method that leverages PPDAC for data augmentation, against all baseline models in an experimental setup comprising only 16 training examples. In this rigorous test, FewshotT5-PPDAC attains remarkable F1 scores of 78.4%, 51.4%, 58.5%, 52.2%, 43.4%, and 55.5% across six distinct datasets within Split I of the MRQA 2019 shared task. These scores represent substantial improvements over the leading baselines by margins of 11.3%, 0.8%, 11.8%, 13.3%, 3.6%, and 10.4%, respectively, underscoring the efficacy of PPDAC in amplifying question-answering performance in constrained data environments. Notably, when compared against the advanced KECP method in Figure 2, our approach demonstrates a commanding lead of 11.3%, 26.6%, 26.6%, 23.8%, 8.1%, and 21.6% across the six datasets, respectively. While our primary goal was to apply our proposed components within the KECP framework to validate their effectiveness, the lack of open-source availability for KECP constrained us to using it as a baseline for performance comparisons only.

To extend our investigation into the PPDAC's impact on baseline models, we integrated our component across a diverse array of baselines, observing performance across varying training sample sizes. The outcomes, detailed in Table 2, reveal a consistent trend of performance enhancement with PPDAC, particularly pronounced in smaller sample sizes. It's important to note that Table 2 showcases only a subset of our experimental data for clarity and brevity. The comprehensive dataset, encompassing all baseline models and the entire range of training sample sizes, is extensive and therefore has been included in the appendix for those interested in a more detailed exploration. This approach ensures that readers can grasp the immediate benefits of PPDAC from the main text while providing access to the full breadth of our findings for deeper analysis.

This observation is critical, highlighting PPDAC's role in not only boosting EQA performance but also in reducing the variability of model outcomes, as evidenced by a decreased standard deviation among the enhanced baselines. Notably, our analysis identified instances where performance gains were less marked, particularly in datasets characterized by complex questions that exhibit significant syntactic or lexical variance from their answers. This suggests a potential limitation of data augmentation strategies in handling the intricate semantics of complex questions, a factor that merits further exploration to optimize the efficacy of augmentation techniques in such contexts.

A pivotal finding of our study is the pronounced benefit of PPDAC in scenarios with limited data, a testament to its utility in enhancing the robustness and reliability of EQA models under the few-shot paradigm. While the performance improvements are more modest in settings with larger data volumes, the reduction in standard deviation after the application of the PPDAC plugin across most baseline models highlights an increased stability and consistency in model performance.

In summary, our research underscores the transformative potential of PPDAC in bolstering the capabilities of few-shot EQA methods. By facilitating more effective data augmentation, PPDAC not only enhances model performance across a spectrum of training conditions but also contributes to a deeper understanding of the dynamics at play in few-shot learning environments. These insights pave the way for future advancements in EQA, particularly in the development of more resilient and adaptable models suited to the diverse challenges inherent in real-world applications.

4.5 The Efficiency of PPDAC

In addition to our comprehensive performance evaluation, we also assessed the time and space efficiency of the PPDAC to demonstrate its practicality in real-world applications. Given that PPDAC operates as a pre-processing step prior to the fine-tuning of pre-trained models, it introduces minimal overhead, ensuring that the efficiency of the fine-tuning process remains largely unaffected. This is a crucial advantage, as it allows for the significant enhancements in EQA performance without imposing substantial additional computational costs.

To quantify the impact of PPDAC, we measured the additional time and space required for data aug-

#Train Samples	Training time	Evaluation time
Splinter	25s	46.78s
Splinter-PPDAC	64s	47.37s
FewshotBART	48s	148.3s
FewshotBART-PPDAC	123s	151.2s
FewshotT5	82s	124.2s
FewshotT5-PPDAC	230s	125.6s

Table 3: The table presents a comprehensive comparison of the training and testing times for three models: Splinter, Fewshot-BART, and Fewshot-T5, both before and after the application of the PPDAC component.

#Train Samples	16	32	128
Splinter-PPDAC	**58.2%**	**64.3%**	**73.8%**
Splinter-PPDAC(w/o. PF)	56.4%	62.2%	72.9%
Splinter-PPDAC(w/o. PS)	56.1%	60.6%	73.2%
Splinter-PPDAC(w/o. PF & PS)	55.6%	60.1%	72.7%
Splinter	54.6%	59.2%	72.7%
FewshotBART-PPDAC	**57.4%**	**60.4%**	**71.6%**
FewshotBART-PPDAC(w/o. PF)	56.2%	57.9%	68.9%
FewshotBART-PPDAC(w/o. PS)	56.6%	58.2%	69.1%
FewshotBART-PPDAC(w/o. PF & PS)	56.0%	57.3%	68.6%
FewshotBART	55.5%	56.8%	68.0%
FewshotT5-PPDAC	**78.4%**	**80.9%**	**86.3%**
FewshotT5-PPDAC(w/o. PF)	61.5%	66.4%	75.5%
FewshotT5-PPDAC(w/o. PS)	59.6%	67.1%	77.5%
FewshotT5-PPDAC(w/o. PF & PS)	55.7%	63.5%	72.6%
FewshotT5	47.8%	56.6%	64.6%

Table 4: The table presents the ablation study results on the SQuAD1.1 dataset for PPDAC in a few-shot learning context. The notation (w/o. PF) signifies the removal of the Paraphrase Filter from PPDAC, highlighting its impact. Similarly, (w/o. PS) indicates the exclusion of the Paraphrase Selector, allowing assessment of its individual contribution. Lastly, (w/o. PF & PS) refers to the scenario where both Paraphrase Filter and Paraphrase Selector are removed, providing insights into their combined effects on enhancing model performance.

mentation and compared it to the baseline fine-tuning process without PPDAC. As evident from Table 3, while there is an observable increase in training time during the fine-tuning process, this increase is attributable to the data augmentation, with the enhanced data volume reaching 2 to 3 times that of the original dataset, and remains within an acceptable range. Notably, the model evaluation time is unaffected by the data augmentation component. The extra time needed for the data augmentation process primarily depends on the volume of the original data to be augmented. The additional space requirements for our module arise from the deployment of the paraphrase and filter models. The paraphrase model itself comprises 223 M parameters. In contrast, the adequacy filter and fluency filter are characterized by their parameter volumes of 109.5 M and 369 M, respectively. Overall, the increase in time and space requirements after applying the PPDAC component is marginal, reinforcing PPDAC's utility as a resource-efficient component in the EQA model training workflow.

4.6 Ablation Study

To elucidate the individual contributions of the Plug-and-Play Data Augmentation Component (PPDAC) to enhancing few-shot Extractive Question Answering (EQA) performance, we meticulously conducted an ablation study. This investigation, now encompassing both the paraphrase selector (PS) module and the Paraphrase Filter (PF) module, was performed across varying data scales—16, 64, and 128 shots within the SQuAD1.1 dataset. The paraphrase selector (PS) module and the Paraphrase Filter (PF) module are integral to enhancing the model's performance by ensuring the generated paraphrases not only retain semantic fidelity to the original questions but also meet quality standards in terms of relevance and coherence. The PS module is crucial for selecting the most appropriate paraphrases by evaluating their alignment with the original question, whereas the PF module filters out paraphrases that do not meet predetermined quality criteria, thereby streamlining the data augmentation process for optimized few-shot learning outcomes. The findings, detailed in Table 4, are pivotal for understanding the nuanced impact of each component.

1) **Splinter-PPDAC(w/o. PS/PF)**: This variant employs the Splinter model integrated with PPDAC, excluding either the PS or PF module. This modification is designed to individually assess the impact of removing each module on the model's performance.

2) **FewshotBART-PPDAC(w/o. PS/PF)**: Similarly, this setup uses the FewshotBART framework with PPDAC, removing either the PS or PF module to evaluate their individual contributions to the model's architecture and effectiveness.

3) **FewshotT5-PPDAC(w/o. PS/PF)**: Following the same approach, this configuration employs the FewshotT5 framework with PPDAC, omitting either the PS or PF module to explore their separate roles in improving model performance.

From Table 3, it is evident that the performance of all models declines universally when any one components are removed, the simultaneous exclusion of both the Paraphrase Filter (PF) and Paraphrase Selector (PS) from PPDAC results in the most pronounced performance deterioration. For instance, within the FewshotT5-PPDAC model using 16 training examples, the removal of PF and PS modules led to performance drops of 16.9% and 18.8%, respectively, with their combined removal causing a 22.7% reduction. Despite these setbacks, the modified models continue to outperform their original configurations across all training sizes. Crucially, eliminating the PF module typically results in greater performance degradation than removing the PS, leading to the conclusion that the PF module's role in ensuring the quality of paraphrase generation is more critical than the PS component's contribution to selecting optimal paraphrases. This underscores the PF module's paramount importance in the enhancement of few-shot EQA performance through quality control over quantity.

Despite these performance decrements, models stripped of both PS and PF, relying solely on the parrot generator component, still surpassed baseline models. This underscores the inherent value of the PPDAC framework in improving EQA outcomes, even in its most reduced form. Among the components, the PS module emerges as particularly crucial, significantly impacting model performance by optimizing paraphrase selection for training efficiency and accuracy.

This detailed analysis not only highlights the indispensable contributions of the PS and PF modules to the PPDAC's efficacy but also sets a foundation for further exploration into refining EQA methodologies within few-shot learning paradigms.

5 Conclusion

In this work, we introduce a novel Plug-and-Play Data Augmentation Component (PPDAC) designed to enhance the training of pre-trained language models for question-answering (QA) tasks. By generating paraphrases (q') of the original questions within (question, passage, answer) triples, PPDAC significantly enriches the training dataset, creating additional (q', passage, answer) combinations. This innovative approach is particularly beneficial in few-shot learning scenarios, where data scarcity often hampers model performance. Our comprehensive experiments across various QA baselines demonstrate the efficacy of PPDAC, with notable improvements observed even in extremely data-constrained environments. Specifically, in a 16-shot learning setup on the SQuAD dataset, PPDAC facilitated performance enhancements of 3.6%, 1.9%, and 30.6% for the Splinter, FewshotBART, and FewshotT5 models, respectively. Remarkably, the integration of PPDAC with FewshotT5 not only achieved significant gains over baseline models but also surpassed existing state-of-the-art methods in few-shot settings by a substantial margin.

Furthermore, our detailed ablation studies shed light on the critical roles of PPDAC's components, particularly the Paraphrase Filter (PF) and Paraphrase Selector (PS). The studies reveal that removing any component leads to a universal decline in model performance, with the simultaneous exclusion of both PF and PS resulting in the most pronounced performance drops. However, even in the absence of these components, the models equipped with the remaining paraphrase generation capabilities of PPDAC continued to outperform their original configurations. This underscores the transformative impact of PPDAC on enhancing QA model performance in few-shot learning contexts, with the PF module, in particular, emerging as a key contributor by prioritizing quality over quantity in paraphrase generation.

Looking ahead, we acknowledge the effectiveness of large language models (LLMs) in data augmentation. However, the innovation of our approach lies in its novel framework and process. While our current implementation utilizes a specific paraphrase generation model, we recognize its adaptability. In future iterations, we plan to incorporate the latest LLMs for paraphrase generation, emphasizing our innovative approach tailored for question-answering tasks.

The Plug-and-Play Data Augmentation Component (PPDAC) currently enhances question paraphrasing within the (question, passage, answer) framework. Future research may expand this approach to include passage paraphrasing, thereby enriching the training dataset and establishing a more solid foundation for fine-tuning pre-trained language models. This expansion aims to advance few-shot learning in question answering, steering towards more sophisticated and effective natural language understanding systems.

Acknowledgements

This work was supported in part by the NSFC (No. 62266023), the Science and Technology Research Project of the Department of Education of Jiangxi Province (GJJ210325, GJJ2200354). We would like to thank anonymous reviewers for the insightful comments. Mingwen Wang is the corresponding author.

References

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Jianfeng Gao, Songhao Piao, Ming Zhou, et al. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *International conference on machine learning*, pages 642–652. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Or Castel, Ori Ram, Avia Efrat, and Omer Levy. 2021. How optimal is greedy decoding for extractive question answering? *arXiv e-prints*, pages arXiv–2108.

Rakesh Chada and Pradeep Natarajan. 2021. Fewshotqa: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6081–6090.

Prithiviraj Damodaran. 2021. Parrot: Paraphrase generation for nlu. *v1.0*.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. Mrqa 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13.

Michael Glass, Alfio Gliozzo, Rishav Chakravarti, Anthony Ferritto, Lin Pan, GP Shrivatsa Bhargav, Dinesh Garg, and Avirup Sil. 2020. Span selection pre-training for question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2773–2782.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Aniruddha Kembhavi, Minjoon Seo, Dustin Schwenk, Jonghyun Choi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pages 5376–5384. Institute of Electrical and Electronics Engineers Inc.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

DP Kingma. 2014. Adam: a method for stochastic optimization. In *Int Conf Learn Represent*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. 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.

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. 2021. Few-shot question answering by pretraining span selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3066–3079.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, et al. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. *BMC Bioinformatics*, 16(1):138.

J. Wang, Chengyu Wang, Minghui Qiu, Qiuhui Shi, Hongbin Wang, Jun Huang, and Ming Gao. 2022. Kecp: Knowledge enhanced contrastive prompting for few-shot extractive question answering. In *Conference on Empirical Methods in Natural Language Processing*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ningyu Zhang, Shumin Deng, Xu Cheng, Xi Chen, Yichi Zhang, Wei Zhang, Huajun Chen, and Hangzhou Innovation Center. 2021. Drop redundant, shrink irrelevant: Selective knowledge injection for language pretraining. In *IJCAI*, pages 4007–4014.

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16 Examples								
RoBERTa	7.7±4.3	7.5±4.4	17.3±3.3	1.4±0.8	6.9±2.7	10.5±2.5	16.7±7.1	3.3±2.1
RoBERTa-PPDAC	**10.1±1.1**	5.7±1.4	**19.1±2.8**	**5.9±0.6**	**8.1±1.8**	**11.0±2.0**	**19.5±1.0**	3.2±1.5
SpanBERT	12.5±5.7	12.8±5.4	19.7±3.6	6.0±1.6	13.0±4.2	12.6±3.6	22.0±4.6	5.6±2.5
SpanBERT-PPDAC	**20.2±6.9**	**16.3±2.1**	**20.7±3.7**	**11.9±0.4**	**22.1±1.6**	**13.4±0.8**	**23.7±0.6**	**7.8±1.2**
Splinter	54.6±6.4	18.9±4.1	27.4±4.6	20.8±2.7	26.3±3.9	24.0±5.0	28.2±4.9	19.4±4.6
Splinter-PPDAC	**58.2±2.5**	**21.4±4.8**	26.9±2.2	19.3±2.2	**28.7±2.1**	**28.8±2.6**	**28.8±1.0**	19.4±3.9
FewshotBART	55.5±2.0	50.5±1.0	46.7±2.3	38.9±0.7	39.8±0.04	45.1±1.5	49.4±0.02	19.9±1.25
FewshotBART-PPDAC	**57.4±0.8**	47.1±0.1	**47.7±0.3**	37.8±0.6	**45.9±1.2**	**60.4±1.1**	**49.9±1.8**	**20.1±1.0**
FewshotBARTL	68.9±2.7	65.2±1.8	60.4±2.0	48.4±2.2	47.8±5.4	58.0±1.8	63.0±1.1	37.7±3.7
FewshotBARTL-PPDAC	**73.2±1.5**	59.6±3.2	**60.5±2.4**	**49.2±0.8**	**59.5±1.0**	**59.3±2.4**	**64.6±0.8**	37.4±1.3
FewshotT5	47.8±6.9	50.6±4.9	28.5±14.5	26.8±2.7	37.0±3.3	44.9±3.5	46.3±5.9	25.9±5.0
FewshotT5-PPDAC	**78.4±0.2**	**51.4±2.0**	**58.5±1.0**	**52.2±0.5**	**43.4±2.1**	**55.5±0.3**	**58.6±0.4**	**32.1±2.5**
KECP	67.1±4.7	24.8±2.4	31.9±2.2	28.4±1.9	35.3±2.4	33.9±2.0	-	-
32 Examples								
RoBERTa	18.2±5.1	10.5±1.8	22.9±0.7	3.2±1.7	13.5±1.8	10.4±1.9	23.3±6.6	4.3±0.9
RoBERTa-PPDAC	**26.4±1.3**	**10.6±1.6**	**23.8±1.6**	**8.9±0.9**	**15.6±1.0**	**11.8±1.2**	**27.6±1.2**	**4.9±0.2**
SpanBERT	19.0±4.6	19.0±4.8	23.5±0.9	7.5±1.3	20.1±3.9	14.4±2.9	32.5±3.5	7.4±1.1
SpanBERT-PPDAC	**39.9±2.9**	**20.3±0.7**	**28.1±1.6**	**17.4±0.8**	**22.3±2.1**	**21.9±3.3**	**36.1±4.1**	**19.0±3.5**
Splinter	59.2±2.1	28.9±3.1	33.6±2.4	27.5±3.2	34.8±1.8	34.7±3.9	36.5±3.2	27.6±4.3
Splinter-PPDAC	**64.3±1.1**	**28.8±3.5**	**34.6±1.2**	**27.7±2.1**	**35.6±1.2**	**39.5±1.2**	**37.0±0.8**	**27.7±3.5**
FewshotBART	56.8±2.1	52.5±0.7	50.1±1.1	40.4±1.5	41.8±0.02	47.9±1.4	52.3±0.02	22.7±2.3
FewshotBART-PPDAC	**60.4±1.9**	51.3±1.5	50.1±0.9	**41.6±0.3**	**49.7±0.5**	**61.5±0.7**	**56.3±2.6**	**26.7±2.0**
FewshotBARTL	72.3±1.0	65.1±1.2	61.5±1.7	51.7±1.7	58.3±1.5	60.4±0.2	67.8±1.0	37.7±9.8
FewshotBARTL-PPDAC	**76.0±1.0**	58.1±2.2	**62.2±0.7**	**52.5±0.7**	**63.5±0.9**	**62.2±0.7**	**70±0.4**	**44.1±0.3**
FewshotT5	56.6±1.5	50.2±9.0	37.5±12.5	33.2±4.6	48.4±5.6	53.6±1.4	57.7±4.2	29.8±2.6
FewshotT5-PPDAC	**80.9±0.4**	50.2±1.1	**60.1±0.1**	**55.6±0.8**	**48.5±1.3**	**57.9±0.5**	**63.8±0.1**	**39.6±3.2**
64 Examples								
RoBERTa	28.4±1.7	12.5±1.4	24.2±1.0	4.6±2.8	19.8±2.4	15.0±3.9	34.0±1.8	5.4±1.1
RoBERTa-PPDAC	**36.3±1.4**	**15.6±0.8**	**26.4±1.1**	**17.3±0.3**	**20.2±1.2**	**18.7±1.2**	**37.5±1.7**	**6.7±0.5**
SpanBERT	33.6±4.3	22.8±2.6	28.4±1.8	8.8±2.4	26.7±2.9	21.8±1.5	43.9±4.5	7.4±1.2
SpanBERT-PPDAC	**49.8±1.6**	**23.0±1.2**	**33.7±1.0**	**26.3±2.6**	**28.6±1.5**	**35.1±1.7**	**50.5±1.6**	**25.9±2.3**
Splinter	65.2±1.4	35.5±3.7	38.2±2.3	37.4±1.2	39.8±3.6	45.4±2.3	49.5±3.6	35.9±3.1
Splinter-PPDAC	**66.6±0.4**	**36.1±2.1**	**38.4±0.3**	**37.8±1.4**	**41.2±1.3**	**47.6±1.7**	**50±0.7**	**36.1±0.8**
FewshotBART	61.5±2.3	50.8±2.2	53.0±0.5	42.7±2.2	46.1±2.9	51.2±1.0	61.8±2.8	27.6±1.8
FewshotBART-PPDAC	**67.8±0.6**	**53.5±0.6**	51.3±0.7	**44.1±0.3**	**55.2±1.3**	**64.4±0.3**	**63.8±1.0**	**34.3±0.4**
FewshotBARTL	73.6±1.9	64.6±1.4	63.0±2.1	53.5±0.9	65.5±2.4	62.9±1.6	73.9±0.8	45.0±1.7
FewshotBARTL-PPDAC	**78.6±0.6**	55.6±0.6	63.3±0.8	**53.7±0.6**	**67.6±0.7**	**63.3±0.8**	72.0±1.6	**46.2±0.9**
FewshotT5	57.2±5.6	52.4±5.9	48.6±2.1	40.2±4.1	54.4±3.0	56.3±2.9	63.8±2.5	32.1±2.7
FewshotT5-PPDAC	**83.5±0.2**	50.8±0.7	**62.5±0.7**	**58.8±0.8**	**54.4±0.2**	**61.4±0.3**	**68.7±1.4**	**47.5±0.9**
128 Examples								
RoBERTa	43.0±7.1	19.1±2.9	30.1±1.9	16.7±3.8	27.8±2.5	27.3±3.9	46.1±1.4	8.2±1.1
RoBERTa-PPDAC	**45.2±4.8**	**19.2±1.1**	**33.4±1.1**	**28.0±1.4**	**27.9±0.8**	**30.2±0.7**	**49.4±2.5**	**9.0±0.6**
SpanBERT	48.5±7.3	24.2±2.1	32.2±3.2	17.4±3.1	34.3±1.1	35.1±4.2	55.3±3.8	9.4±3.0
SpanBERT-PPDAC	**53.9±3.0**	19.3±1.8	**34.1±1.5**	**27.3±3.6**	**35.1±0.9**	**39.8±1.0**	**57.5±0.5**	**27.8±2.1**
Splinter	72.7±1.0	44.7±3.9	46.3±0.8	43.5±1.3	47.2±3.5	54.7±1.4	63.2±4.1	42.6±2.5
Splinter-PPDAC	**73.8±0.7**	**45.1±1.0**	**48.8±0.4**	**43.6±0.9**	**48.0±2.1**	**55.3±0.5**	**64.4±2.0**	37.9±2.0
FewshotBART	68.0±0.3	50.1±1.8	53.9±0.9	47.9±1.2	58.1±1.4	54.8±0.8	68.5±1.0	29.7±2.4
FewshotBART-PPDAC	**71.6±1.0**	**53.9±1.2**	**54.1±0.2**	**49.9±0.08**	**59.1±0.2**	**67.1±0.6**	**72.4±0.2**	**37.8±0.4**
FewshotBARTL	79.4±1.5	65.8±0.9	64.3±1.3	57.0±0.9	67.7±1.0	75.1±1.5	75.0±1.5	48.4±2.7
FewshotBARTL-PPDAC	**79.9±0.5**	57.5±0.8	**65.4±0.8**	**57.2±1.0**	**68.1±0.5**	65.4±0.8	**77.2±0.9**	**50.5±2.1**
FewshotT5	64.6±6.1	51.7±3.1	47.0±4.6	40.0±1.9	57.0±4.5	56.1±3.7	68.2±3.6	33.6±2.1
FewshotT5-PPDAC	**86.3±0.04**	**52.8±0.3**	**64.3±0.1**	**60.9±0.4**	**60.8±0.4**	**64.4±0.9**	**73.9±0.9**	**51.5±0.8**

Table 5: The table spans training examples from 16 to 128, where data highlighted in bold indicates performance improvements post the application of the PPDAC component.

A Complete Experimental Data Tables

The appendix presents the complete set of experimental data tables, which, due to their extensive length, are divided into two subsets for 16, 32, 64, 128 and 256, 512, 1024,which are respectively Table 5 and Table 6. In these tables, improvements in model performance following the application of PPDAC are highlighted in bold font. The baseline KECP data is included for comparative reference as per the original publication, given the absence of open-source code from the authors. The term ”FewshotBARTL” denotes the large version of the BART model within the Fewshot framework. Given its significantly larger parameter volume compared to BART-base and T5-base, it is included in the tables solely for comparative reference.

Model	**SQuAD**	**TriviaQA**	**NQ**	**NewsQA**	**SearchQA**	**HotpotQA**	**BioASQ**	**TextbookQA**
256 Examples								
RoBERTa	56.1±5.2	26.9±3.5	36.0±3.2	31.2±2.4	37.5±1.7	42.7±3.1	63.5±1.8	13.5±1.9
RoBERTa-PPDAC	**56.2±2.8**	26.9±2.1	**36.1±1.1**	**32.0±1.4**	**37.9±0.8**	42.2±0.7	63.4±2.5	**14.3±0.6**
SpanBERT	55.2±8.8	34.0±5.7	41.3±2.2	34.7±4.1	42.3±4.1	49.4±4.0	67.5±3.9	18.2±4.5
SpanBERT-PPDAC	**55.9±3.0**	**34.1±1.8**	**42.1±1.5**	34.3±3.6	42.1±0.9	**49.8±1.0**	67.5±0.5	**31.6±2.1**
Splinter	76.8±0.6	57.2±2.2	54.6±1.2	49.0±0.4	55.7±1.9	62.0±1.6	77.4±2.0	48.5±2.2
Splinter-PPDAC	76.8±0.5	**57.3±1.0**	**54.8±0.4**	**49.6±0.9**	**56.0±2.1**	**63.3±0.5**	77.4±2.0	**48.9±2.0**
FewshotBART	74.3±0.2	52.7±0.4	56.9±0.3	53.4±0.4	65.4±0.5	59.4±0.5	81.0±0.2	44.6±0.8
FewshotBART-PPDAC	**74.6±0.2**	52.7±0.1	**57.9±0.2**	**53.8±0.4**	**64.7±0.1**	59.4±0.5	**82.1±0.7**	**47.2±0.9**
FewshotBARTL	81.9±1.3	56.2±1.5	66.4±0.8	59.6±0.6	71.1±0.7	67.1±0.2	82.9±0.3	54.4±0.2
FewshotBARTL-PPDAC	**82.2±0.04**	**61.6±0.6**	**66.6±0.1**	59.6±0.3	**71.4±0.4**	**67.3±0.1**	**83.6±1.2**	**57.2±0.3**
FewshotT5	87.4±0.3	55.4±0.7	65.6±0.1	61.8±0.1	65.9±0.4	66.5±0.4	77.9±0.1	50.3±1.5
FewshotT5-PPDAC	**87.5±0.2**	**55.9±1.0**	**65.8±0.3**	**62.3±0.4**	**66.0±0.2**	66.3±0.4	**80.7±0.09**	**55.0±1.5**
512 Examples								
RoBERTa	67.3±0.7	38.7±3.8	46.7±2.2	41.5±2.2	46.9±1.6	56.7±1.3	77.0±1.9	27.0±2.2
RoBERTa-PPDAC	67.2±0.5	38.7±2.8	46.7±1.2	41.5±1.2	46.9±0.6	56.0±0.8	**77.3±0.9**	**27.5±1.2**
SpanBERT	70.0±4.3	44.2±2.9	51.5±1.8	42.4±2.6	53.9±3.2	61.6±1.7	80.3±3.0	33.7±3.4
SpanBERT-PPDAC	**70.9±2.0**	44.2±1.9	51.5±1.0	42.4±1.6	53.9±2.2	**61.7±0.7**	**80.4±1.0**	**34.0±1.4**
Splinter	80.1±0.4	61.9±1.8	61.4±1.1	53.2±0.9	63.1±1.6	66.2±0.6	84.8±0.9	54.2±1.7
Splinter-PPDAC	**80.8±0.7**	61.9±0.8	61.4±1.0	53.2±0.5	63.1±0.6	**66.4±0.6**	**85.0±0.9**	**54.5±0.7**
FewshotBART	77.1±0.2	51.9±0.6	59.3±0.1	56.4±0.3	67.3±0.2	63.5±0.7	87.6±0.8	50.4±0.04
FewshotBART-PPDAC	**77.2±0.3**	**52.8±0.3**	**60.1±0.09**	**56.5±0.1**	**67.7±0.2**	63.5±0.6	87.6±0.3	**54.9±0.1**
FewshotBARTL	83.3±0.8	60.4±1.4	66.4±0.8	60.8±0.09	73.0±0.4	70.3±0.4	84.4±0.1	55.7±1.2
FewshotBARTL-PPDAC	83.1±0.1	**63.2±0.5**	**67.4±0.2**	**61.9±0.2**	72.8±0.8	69.6±0.8	**86.9±0.3**	**57.5±0.5**
FewshotT5	87.9±0.1	57.7±0.2	67.8±0.2	63.5±0.08	69.6±0.1	68.8±0.4	83.0±0.1	56.7±0.6
FewshotT5-PPDAC	**88.5±0.1**	**58.1±0.4**	**68.0±0.4**	**63.9±0.1**	**69.8±0.1**	68.8±0.4	**84.1±0.9**	**58.8±0.1**
1024 Examples								
RoBERTa	73.8±0.8	46.8±0.9	54.2±1.1	47.5±1.1	54.3±1.2	61.8±1.3	84.1±1.1	35.8±2.0
RoBERTa-PPDAC	73.2±0.8	**47.1±1.1**	**54.6±1.1**	47.3±1.4	54.1±0.8	**62.0±0.7**	**84.3±1.0**	**36.0±0.6**
SpanBERT	77.8±0.9	50.3±4.0	57.5±0.9	49.3±2.0	60.1±2.2	67.4±1.6	89.3±0.6	42.3±1.9
SpanBERT-PPDAC	**77.9±0.3**	**50.4±1.8**	**58.5±0.5**	**50.1±3.6**	**60.5±0.9**	67.3±1.0	89.3±0.6	42.1±2.1
Splinter	82.8±0.8	64.8±0.9	65.5±0.5	57.3±0.8	67.3±1.3	70.3±0.8	91.0±1.0	54.5±1.5
Splinter-PPDAC	82.8±0.7	**64.9±1.0**	**66.2±0.4**	57.0±0.9	67.3±1.1	**70.4±0.5**	91.0±0.5	**54.7±1.3**
FewshotBART	79.6±0.09	55.5±0.7	61.4±0.1	58.9±0.6	70.0±0.04	65.8±0.7	92.2±0.8	50.0±0.2
FewshotBART-PPDAC	**79.7±0.1**	55.5±0.9	**62.4±0.1**	58.7±0.1	**70.5±0.2**	65.0±0.6	**92.5±0.2**	**51.8±0.4**
FewshotBARTL	83.0±0.2	62.1±0.1	68.2±0.2	62.2±0.4	73.4±0.8	66.5±0.1	91.0±0.3	52.5±0.5
FewshotBARTL-PPDAC	**84.5±0.5**	**64.1±0.4**	**69.0±0.4**	**63.6±0.2**	**74.4±0.6**	**71.5±0.2**	**91.7±0.2**	**58.9±0.4**
FewshotT5	88.6±0.2	60.0±0.5	69.4±0.3	64.7±0.2	73.0±0.2	70.2±0.2	86.9±0.2	59.7±0.5
FewshotT5-PPDAC	**89.0±0.09**	**61.0±0.08**	69.4±0.08	**64.9±0.1**	**73.1±0.1**	**71.2±0.2**	**87.8±0.1**	**60.1±0.6**

Table 6: This table encompasses training examples ranging from 256 to 1024, with data shown in bold representing performance enhancements following the incorporation of the PPDAC component.