GOTTA: Generative Few-shot Question Answering by Prompt-based Cloze Data Augmentation

Xiusi Chen* Yu Zhang† Jinliang Deng‡ Jyun-Yu Jiang§ Wei Wang¶

Abstract

Few-shot question answering (QA) aims at precisely discovering answers to a set of questions from context passages while only a few training samples are available. Although existing studies have made some progress and can usually achieve proper results, they suffer from understanding deep semantics for reasoning out the questions. In this paper, we develop GOTTA, a **G**enerative pr**O**mp**T**-based da**T**a **A**ugmentation framework to mitigate the challenge above. Inspired by the human reasoning process, we propose to integrate the cloze task to enhance few-shot QA learning. Following the recent success of prompt-tuning, we present the cloze task in the same format as the main QA task, allowing the model to learn both tasks seamlessly together to fully take advantage of the power of prompt-tuning. Extensive experiments on widely used benchmarks demonstrate that GOTTA consistently outperforms competitive baselines, validating the effectiveness of our proposed prompt-tuning-based cloze task, which not only fine-tunes language models but also learns to guide reasoning in QA tasks. Further analysis shows that the prompt-based loss incorporates the auxiliary task better than the multi-task loss, highlighting the strength of prompt-tuning on the few-shot QA task.

Keywords

question answering, knowledge base, entity, data augmentation

1 Introduction

Question answering (QA) is the task of precisely discovering answers to natural language questions given the narrative contexts. With a wide range of downstream applications, such as knowledge graph completion [22], response recommendation [46], review opinion mining [45], and product attribute extraction [39], it has drawn a lot of attention in the text mining community and has risen to one of the holy-grail tasks. Following the line of supervised learning, one can successfully build QA methods that achieve decent results. However, the assumption that a large amount of annotated QA training examples quickly poses limitations since annotation requiring efforts from domain experts is extremely expensive.

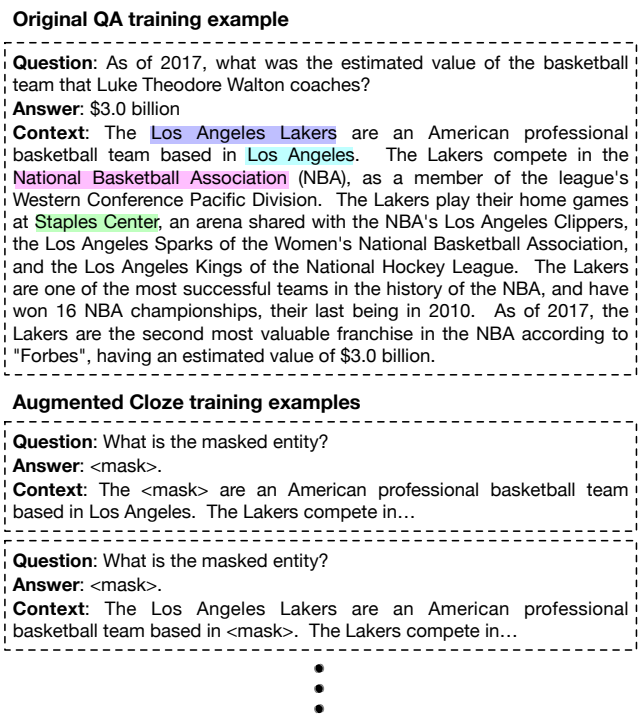

Figure 1: An example of how entity-aware text masking and prompt-style data augmentation work. GOTTA selects entities that are covered by knowledge bases, and creates prompt-style augmented data for training purpose.

We investigate the few-shot QA task, which aims to solve the QA task while only a few training examples are present. Under the few-shot setting, most existing approaches either propose a new task and pre-train a large language model from scratch [29], or fine-tune the pre-trained model on the training examples [5]. These practices do not explicitly understand the entities in the input text (i.e., the context text and the question text) before generating the output (i.e., the answer

*University of California, Los Angeles. xchen@cs.ucla.edu
†University of Illinois at Urbana-Champaign. yuz9@illinois.edu
‡Univ of Technology Sydney. jinliang.deng@student.uts.edu.au
§Amazon Search. jyunyu@amazon.com
¶University of California, Los Angeles. weiwang@cs.ucla.edu

text), which contradicts the conventional human thinking process. For example, in reading comprehension exams, people have to fully understand and digest the context semantics before getting precise answers. In other words, directly mapping from the text and the question to the answer lacks a deep understanding of the context.

To bridge this gap, we develop GOTTA, a **G**enerative pr**O**mp**T**-based da**T**a **A**ugmentation framework for few-shot QA. In GOTTA, we design a knowledge-based cloze task to serve as a companion to enhance the main QA task. To make the cloze task more dedicated for QA, we utilize publicly available knowledge bases and focus on the covered entities by only selecting the entities in the text as the object to construct cloze problems. By constructing more data for fine-tuning, we incorporate the external knowledge in the knowledge bases in the hope of introducing more inductive bias that is beneficial to the QA task. The inductive bias provides extra supervision beyond the weak supervision signals only provided in the few-shot QA training set. Intuitively, the cloze task is to imitate the human behavior of understanding the context by filling in the blanks. We conduct this entity-aware cloze because identifying the entities and understanding their relations is crucial for solving QA problems on the same chunk of text.

Inspired by recent advantages of prompt-tuning, as shown in Figure 1 and Figure 2, we feature both QA and cloze tasks in the same prompt template to align with each other at the pre-training stage. Following this routine, no redundant model parameters are introduced while the pre-trained model can maximize the performance on our downstream QA task, especially under the few-shot setting. Although our cloze task is quite similar to the popular masked language modeling (MLM), there are two major distinctions between entity-aware text masking and MLM. First, MLM randomly masks word tokens while entity-aware text masking only targets entities that are more likely to be relevant to the QA task. Second, MLM is usually pipelined with a softmax function to select one token while entity-aware text masking generates a token sequence to form a text span, which is more favored by QA tasks. Extensive experiments on publicly available and conventional benchmarks demonstrate that GOTTA is able to achieve generally better results over competitive baselines, validating the effectiveness of the cloze task. Further in-depth analysis shows that the prompt-based loss incorporates the auxiliary task better than classification loss, highlighting the effectiveness of prompt-tuning on the few-shot QA task.

We summarize our contributions as follows:

- We propose to incorporate the cloze task as a data augmentation module to extract self-supervised training examples to enhance the learning for few-shot QA.
- We formulate both QA and cloze tasks in the same format, allowing us to apply prompt-tuning to take full advantage of pre-trained large language models.
- We conduct extensive experiments on publicly accessible benchmarks to validate the effectiveness of GOTTA, and observe consistent improvement over competitive compared methods. Beyond that, we also study the necessity of different parts of the model, providing the readers with a better understanding of the framework.[1]

2 Related Work

Existing studies most related to our work come from three aspects: few-shot QA, prompt-tuning, and data augmentation. In this section, we briefly recap and distinguish our proposed method from theirs.

Few-Shot QA. Prior studies in QA either reuse the high-performing pre-trained language models (PLMs) [18, 13], or train a model from scratch on synthetic QA data [27, 21, 2]. However, all of them require fine-tuning the models on massive annotated data from the downstream QA task, which is often impractical in real-world cases. Several approaches have recently been developed to allow the model to quickly adapt to the downstream task with solely a handful of annotated data [29, 5]. Ram et al. [29] tailor the pre-training scheme specialized for handling QA tasks. They design a recurring span selection objective for pre-training, which aligns with the common objective in extractive QA tasks. To save the effort to pre-train the model on a large-scale corpus, Chada and Natarajan [5] seek to explore the capacity of the existing PLMs. They propose a simple framework, known as FewshotQA, where a QA-style prompt is constructed to cast the QA problem as a text generation problem. Specifically, the prompt is created as a concatenation of the question and a mask token representing the answer span. In this way, the input format is geared toward processing by the PLMs. Distinct from these two studies, we focus on exploring more relevant information in the context data, aside from the annotated QA pairs, to fine-tune the model under the few-shot setting. KECP [38] is a concurrent work with GOTTA that focuses only on extractive QA (EQA). Also inspired by prompt-tuning, KECP views the EQA task as a non-autoregressive MLM generation problem and uses a span-level contrastive learning objective to improve the final performance.

Prompt-Tuning. Standard fine-tuning of PLMs for few-shot learning does not achieve satisfying performance in many cases because the limited training sam-

[1] The code for GOTTA is at `https://github.com/xiusic/Gotta`.

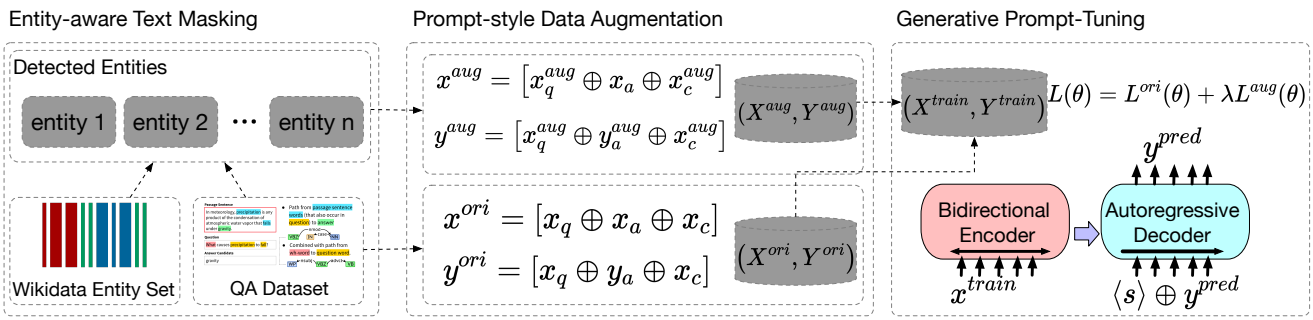

Figure 2: Framework overview for GOTTA.

ples may not be sufficient for optimizing the parameters in the newly introduced task head. To reuse the language modeling capability of PLMs without introducing randomly initialized parameters, prompt-based approaches [10, 12, 25, 26, 30, 32, 33] formulate training samples as natural language prompt templates so that the downstream tasks can be solved as a masked token prediction problem. Further studies propose to replace the manual design of prompts with automatic search or learning [6, 11, 19, 23, 44, 47]. Although prompt-tuning has demonstrated remarkable few-shot learning performance in some tasks (e.g., text classification and natural language inference [37]), it has not been extensively explored in question answering. In this paper, we explore a prompt-based data augmentation framework for few-shot QA.

Data Augmentation. Under the few-shot setting, data augmentation mainly aims to create more training data based on a small number of provided training samples to overcome label scarcity when training the model. Pioneering studies on text data augmentation include EDA [40] and UDA [41], which leverage text editing (e.g., synonym replacement, random swap) and back translation to create more labeled data for text classification. In another line of work, several studies generate training data by fine-tuning autoregressive PLMs on the training set [3, 42] or using label-specific prompts [31] to guide text generation toward the desired label. However, most of the studies mentioned above focus on the task of few-shot text classification. In contrast, our GOTTA framework proposes a cloze-style data augmentation method for few-shot QA.

3 Gotta: The Proposed Framework

The overall framework of GOTTA is illustrated in Figure 2. The core idea is to augment training data with cloze-style questions to force the model to understand the contexts beyond the original questions. We fulfill this idea in three steps: First, we identify the tokens that should be masked in the cloze task. Intuitively, such tokens need to indicate the answers to the original questions. Then, we construct the prompt data by combining the masked tokens and our designed template. Finally, we feed the original QA training samples and the created prompt data into a pre-trained BART model [20] for fine-tuning.

3.1 Entity-aware Text Masking

The first step to fulfill the cloze task is entity-aware text masking. The cloze task is often referred to as "masked language modeling" (MLM) in the literature [7]. Although MLM is widely used as a pre-training task in NLP, it is still less explored how to pick the masked token spans to achieve good performance in a specific downstream task. Indeed, prevailing PLMs like BERT [7] randomly mask a proportion of tokens in each sequence. However, even though PLMs with randomly masking statistically can survive with a large-scale training corpus and the law of large numbers, few-shot QA tasks with only tens of short sequences could potentially receive only weak and noisy samples. For the sake of our QA task, we propose to enable the model to infer the crucial parts of the reasoning procedure. Human reasoning is usually considered as hops between entities [4]. A robust model should be able to recover important masked entities based on their context. To achieve this idea, we take the entity set of the Wikidata knowledge graph [36] as the entity corpus. For each training sample, we extract all the text spans recognized as Wikidata entities. As a result, the created cloze questions will be centered around meaningful entities rather than irrelevant tokens to the QA task, such as articles, pronouns, and stopwords.

3.2 Prompt-style Data Augmentation

Based on the output of entity-aware text masking, we pursue the recent success of prompt-tuning to produce augmented data for the prompt-based GOTTA model. Specifically, we formally formulate the following template to integrate QA and cloze tasks, thereby generating few-shot QA

input data x^{ori} as:

$$x_q = \textit{Question} : \mathbf{q}$$
$$x_a = \textit{Answer} : \langle \text{mask} \rangle$$
$$x_c = \textit{Context} : \mathbf{c}$$
$$x^{ori} = [x_q \oplus x_a \oplus x_c]$$

The labels y are formulated as follows:

$$y_a = \textit{Answer} : \mathbf{a},$$
$$y = [x_q \oplus y_a \oplus x_c],$$

where $\mathbf{q}$, $\mathbf{a}$ and $\mathbf{c}$ are texts of the question, answer text, and context, respectively; $\oplus$ denotes string concatenation.

For the augmented data, we fix the question text $\mathbf{q}^{\text{aug}}$ as follows:

$$\mathbf{q}^{\text{aug}} = \textit{What is the masked entity?}$$

Note that in the augmented cloze data samples, we also mask the selected entity in x_c to form the context text for the augmented data x_c^{aug} in addition to the mask token in x_a. Figure 1 illustrates the details of an augmented data sample (x^{aug}, y^{aug}). Let (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}) denote all the training samples of QA and cloze, respectively. Our complete training set (X^{train}, Y^{train}) is the union of (X^{ori}, Y^{ori}) and (X^{aug}, Y^{aug}).

3.3 Generative Prompt-Tuning One of the most apparent advantages of aligning augmented and original data is the model's capability of seamlessly digesting both without a distinct loss. In a nutshell, GOTTA adopts an encoder-decoder model as:

$$y^{pred} = \textit{decoder}_{\theta_D}(\textit{encoder}_{\theta_E}(x)), \tag{3.1}$$

where θ_E and θ_D are learnable parameters; $x \in X^{train}$ can be either an original training sample or an augmented one.

Our training objective maximizes the log-likelihood of the text in the reference answer $y \in Y^{train}$. The loss functions with respect to the original samples and the augmented samples can be expressed as follows:

$$L^{ori}(\theta) = \sum_{(x,y)\in(X^{ori},Y^{ori})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right) \tag{3.2}$$

and

$$L^{aug}(\theta) = \sum_{(x,y)\in(X^{aug},Y^{aug})} \log\left(\prod_{i=1}^{n} P\left(y_i \mid y_{<i}, x; \theta\right)\right), \tag{3.3}$$

where $\theta = \{\theta_D, \theta_E\}$.

The overall loss function takes a weighted sum:

$$L(\theta) = L^{ori}(\theta) + \lambda L^{aug}(\theta). \tag{3.4}$$

Here, $\lambda > 0$ is a hyperparameter that balances between the QA task and the prompted cloze task.

4 Experiments

In this section, we describe in detail how we set up our experiments, then we report the experimental results and discuss the results. We further provide some in-depth analysis of GOTTA, through which we can better understand the model.

4.1 Experimental Setup Datasets. Following Splinter [29] and FewshotQA [5], we sample subsets from the MRQA 2019 shared task [9] for our few-shot experiments. Specifically, MRQA contains eight widely used benchmark question answering datasets: SQuAD [28], NewsQA [34], TriviaQA [14], SearchQA [8], HotpotQA [43], Natural Questions [17], BioASQ [35], and TextbookQA [15]. Following Splinter [29], smaller training datasets are sampled in a logarithmic manner from the original full datasets, resulting in few-shot datasets with training example numbers 16, 32, 64, and 128.

Comparative Baselines. We evaluate the performance of GOTTA against four competitive few-shot QA methods, including **RoBERTa** [24], **SpanBERT** [13], **Splinter** [29], and **FewshotQA** [5].

Implementation Details. We extract 24,863,792 entities from Wikidata for entity candidate matching. When extracting the entities in the contexts of training samples, we use the Aho-Corasick algorithm[2] [1] to conduct exact multi-pattern lexical matching. For all the models, we use the same hyperparameters during training for a fair comparison. Specifically, the models are optimized by Adam [16] with bias corrections. The learning rate is 2×10^{-5} without learning rate scheduling. The training batch size is set to 2. The maximum sequence length of sequence generation is 100 for FewshotQA and GOTTA. We train all compared models for 25 epochs. The reported results are given by the best-performing checkpoint on the development sets. For GOTTA, we perform a grid search for the loss weight λ in the space $\{0.01, 0.05, 0.1, 0.5, 1.0, 10.0\}$. All the experiments are run on NVIDIA Tesla A100-SXM4 Tensor Core GPUs with 40GB memory.

Evaluation Metrics. Following previous studies [29, 5], we use the F1 score as our evaluation metric. Specifically,

[2] https://github.com/WojciechMula/pyahocorasick/

# examples	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16	336	2,118	883	1,904	2,620	517	591	1,814
32	711	4,287	1,422	2,801	5,452	1,005	1,205	3,934
64	1,539	8,592	2,696	5,867	10,601	2,090	2,568	7,526
128	3,052	17,301	4,989	11,469	21,113	4,128	5,226	15,504

Table 1: Number of augmented training examples per dataset. We construct one training example for each entity extracted from the passages and form the cloze task.

for each sample in the test set, the predicted span and the ground truth answer are treated as bags of words, and F1 scores are applied to compute the overlap between these two sets. If there are multiple ground-truth answers to a particular question, we take the maximum of the corresponding F1 scores.

4.2 Performance Comparison Table 2 shows the few-shot QA performance of compared models across all the benchmarks when 16, 32, 64, and 128 training examples are given. For both **FewshotQA** and **Gotta**, we use BART-large as the backbone PLM. We also report their performance when BART-base is applied as the PLM, in which case the models are denoted as **FewshotQA-base** and **Gotta-base**, respectively. We repeat the same experiment 5 times using different random seeds and report the mean and standard deviation of the results for each method. Furthermore, we include the relative performance gain of GOTTA over the second-best method, i.e., FewshotQA. Overall, GOTTA outperforms all the compared methods by a decent margin in most cases. Even beyond that, we observe a lower variance in results produced by GOTTA over FewshotQA in most cases (24 out of 32), especially when fewer training examples are available (14 out of 16).

Next, let us take a closer look at specific datasets. On SQuAD and HotpotQA, GOTTA consistently achieves higher F1 with lower variance. On TriviaQA, NewsQA, SearchQA, and TextbookQA, we observe relatively more significant performance gains over the best baseline. We conjecture that it is because the number of augmented data samples on these datasets is larger than that on other datasets. Therefore, signals from the cloze task are sufficient to impact the main QA task positively.

4.3 Analysis and Discussions We further provide more in-depth studies to look into which steps and parts contribute the most to GOTTA's performance. Looking back on the design of our model, three key modules are proposed, namely entity masking, prompt data construction, and prompt loss design. Besides, data augmentation also plays a vital role in GOTTA.

4.3.1 Entity Masking We start from the entity masking module. To check whether entity masking benefits the overall performance, we create a variation of GOTTA called GOTTA-random. In GOTTA-random, we remove the entity masking module and randomly mask text spans instead of entities that appear in the Wikidata entity set. As shown in Table 3 comparing between GOTTA and GOTTA-random, we find that: (1) Randomly masking usually yields a higher variance. Although the cloze task can still be fulfilled by randomly selecting phrases, it destabilized the overall QA performance. (2) The full model outperforms the random model in most cases, which validates our hypothesis that masking entities in the context are crucial for selecting the subjects of the cloze examples, thus improving the QA task.

4.3.2 Prompt-tuning vs. Multi-task Learning Prompt data construction is the second key step proposed in our GOTTA framework. As an analysis, we compare prompt tuning with multi-task learning, which can be the other intuitive approach to jointly learn the QA and cloze tasks. Specifically, we denote GOTTA with multi-tasking learning as GOTTA-MTL.

From Table 3, we observe that (1) GOTTA-MTL has apparently worse performance than GOTTA, which validates our claim that formulating the cloze task in the same format of QA is essential. (2) GOTTA-MTL is defeated by GOTTA-what in most cases, meaning that the contribution of prompt is larger than that of entity masking or question text. That being said, aligning the format of QA, cloze along with that of the pre-training task contributes the most to the overall performance.

4.3.3 Question Templates Now that we have shown that it is necessary to formulate the cloze task as prompt-tuning, a natural question is: *Does the question text have an impact on the prompt-tuning performance?* To answer this, we construct another model GOTTA-what to study the effect of question text on the performance. The mere distinct between GOTTA-what and the original model is the question text of the augmented data. Formally, we

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
SpanBERT	18.2±6.7	11.6±2.1	19.6±3.0	7.6±4.1	13.3±6.0	12.5±5.5	15.9±4.4	7.5±2.9
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
FewshotQA-base	55.3±2.7	39.6±6.2	46.9±1.4	36.5±2.6	40.8±4.4	43.7±2.4	52.1±1.6	16.7±2.2
FewshotQA	72.5±3.7	47.1±7.6	57.3±3.2	44.9±4.5	54.3±5.9	59.7±2.2	62.7±4.4	33.1±3.2
GOTTA-base	57.8±2.6	40.8±5.6	47.1±1.1	36.2±1.6	41.8±5.4	45.9±1.7	55.2±2.5	20.5±1.9
GOTTA	**74.6±1.9**	**63.3±8.0**	**58.9±1.9**	**47.3±2.5**	**56.8±3.9**	**59.8±2.1**	**66.1±3.1**	**38.5±5.3**
Improvement%	2.9	34.3	2.8	5.3	4.5	0.1	5.4	16.1
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
SpanBERT	25.8±7.7	15.1±6.4	25.1±1.6	7.2±4.6	14.6±8.5	13.2±3.5	25.1±3.3	7.6±2.3
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
FewshotQA-base	59.5±2.2	50.3±3.1	48.1±2.1	40.7±2.3	49.4±3.2	48.2±1.7	56.7±2.2	24.1±4.2
FewshotQA	73.8±2.2	56.7±5.9	**60.6±2.4**	50.0±2.8	61.4±3.6	61.6±1.5	66.9±4.7	41.7±4.2
GOTTA-base	62.7±1.8	47.7±4.5	49.6±1.3	41.4±2.4	49.8±2.5	49.6±1.3	57.6±3.0	28.1±1.9
GOTTA	**76.0±2.0**	**61.9±4.8**	59.8±2.4	**51.2±1.5**	**63.1±3.1**	**62.7±1.2**	**69.5±1.0**	**46.3±3.7**
Improvement%	3.0	9.1	-1.4	2.4	2.8	1.7	3.8	11.1
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
SpanBERT	45.8±3.3	15.9±6.4	29.7±1.5	12.5±4.3	18.0±4.6	23.3±1.1	35.3±3.1	13.0±6.9
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
FewshotQA-base	66.5±1.1	52.3±2.8	51.5±1.6	43.5±2.0	54.9±2.0	50.7±1.6	64.3±2.3	31.7±2.8
FewshotQA	77.9±2.1	57.9±4.4	60.9±2.5	53.7±1.1	65.4±2.4	63.1±2.2	73.2±3.1	44.8±1.8
GOTTA-base	67.7±0.9	50.6±4.0	51.5±1.3	45.7±1.6	54.6±3.1	52.0±0.8	64.9±2.6	35.5±3.5
GOTTA	**78.9±0.5**	**59.6±1.9**	**63.6±1.0**	**54.3±3.0**	**66.3±2.5**	**64.3±1.7**	**73.2±1.5**	**51.2±2.8**
Improvement%	1.3	3.0	4.4	1.1	1.4	1.9	0.0	14.3
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
SpanBERT	55.8±3.7	26.3±2.1	36.0±1.9	29.5±7.3	26.3±4.3	36.6±3.4	52.2±3.2	20.9±5.1
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
FewshotQA-base	70.8±0.7	45.9±2.1	53.6±1.1	48.4±1.8	58.7±0.9	56.3±0.9	73.8±1.0	37.7±1.1
FewshotQA	78.8±2.7	55.2±1.8	63.3±1.6	56.8±1.1	67.0±1.8	64.9±1.8	77.2±1.5	46.2±5.9
GOTTA-base	71.3±1.3	52.8±2.0	54.2±0.7	49.8±1.6	60.2±1.6	56.3±1.4	73.1±1.9	40.3±3.2
GOTTA	**80.8±1.7**	**60.0±3.6**	**64.9±1.2**	**57.4±1.2**	**69.8±1.5**	**66.7±1.8**	**78.6±2.1**	**53.3±1.7**
Improvement%	2.6	8.8	2.5	1.1	4.3	2.9	1.8	15.3

Table 2: Overall performance in F1 scores across all datasets when the numbers of training examples are 16, 32, 64, and 128. NQ stands for Natural Questions. Improvement% marks the relative performance improvements of GOTTA compared to the best baselines. RoBERTa, SpanBERT, and Splinter have 110M parameters. FewshotQA-base and GOTTA-base have 130M parameters. Both FewshotQA and GOTTA have parameters of size 406M. The average improvements of GOTTA over FewshotQA are significant on all eight datasets in a paired t-test (p-value < 0.05).

change the original question

$$\mathbf{q} = \textit{What is the masked entity?}$$

to the question

$$\mathbf{q} = \textit{What?}$$

Comparing the performance of GOTTA-what with that of GOTTA in Table 3, we observe that the two are comparable. On TriviaQA, GOTTA slightly outperforms GOTTA-what consistently while things are otherwise on any other dataset, with the two going back and forth.

4.3.4 Case Study We further take a look at two concrete test cases. Figure 3 illustrates two examples sampled from the test set of TriviaQA. As we can see, in the left case, both FewshotQA and GOTTA-random generate the incorrect answer *Football*. While this generated answer has highly relevant semantics to the correct answer *2010 FIFA World Cup*, that answer is still not detailed enough. From this observation, we validate our claim that compared with FewshotQA and GOTTA-random without an entity masking module, the full model of GOTTA can generate the answer text in detail from the entity level. In the right case, GOTTA generates the

Model	SQuAD	TriviaQA	NQ	NewsQA	SearchQA	HotpotQA	BioASQ	TextbookQA
16 Examples								
Gotta	**74.6±1.9**	**63.3±8.0**	**58.9±1.9**	**47.3±2.5**	**56.8±3.9**	59.8±2.1	66.1±3.1	38.5±5.3
Gotta-random	72.1±2.6	53.2±8.4	56.2±4.1	46.7±2.2	54.8±6.1	**61.2±1.0**	61.9±2.3	38.4±2.7
Gotta-MTL	71.0±1.9	49.4±7.7	57.8±2.6	45.1±3.5	56.0±5.1	58.4±2.3	62.9±4.5	37.4±3.4
Gotta-what	69.8±2.9	52.0±7.3	57.9±3.3	46.8±1.8	54.9±4.4	60.1±1.0	**66.2±3.3**	**38.8±2.3**
32 Examples								
Gotta	**76.0±2.0**	**61.9±4.8**	59.8±2.4	51.2±1.5	**63.1±3.1**	62.7±1.2	69.5±1.0	**46.3±3.7**
Gotta-random	75.9±2.1	54.7±5.4	59.3±1.7	**51.5±2.2**	62.8±2.3	**63.3±1.6**	67.5±3.8	42.6±4.9
Gotta-MTL	70.9±2.4	55.5±5.8	60.0±1.4	48.7±3.2	60.8±1.7	61.4±1.2	66.7±1.9	41.5±3.6
Gotta-what	74.7±1.1	54.6±5.6	**60.4±2.2**	50.0±1.2	62.4±2.9	60.2±1.7	**70.7±1.3**	40.4±4.1
64 Examples								
Gotta	78.9±0.5	**59.6±1.9**	**63.6±1.0**	**54.3±3.0**	66.3±2.5	**64.3±1.7**	**73.2±1.5**	**51.2±2.8**
Gotta-random	**79.3±1.3**	57.9±3.4	62.2±1.6	53.0±3.0	66.1±3.4	63.8±1.7	72.8±1.7	51.1±3.3
Gotta-MTL	73.9±2.7	54.5±5.0	60.7±1.1	52.6±1.1	65.7±2.3	63.3±1.7	71.6±2.6	45.1±3.3
Gotta-what	78.7±1.2	59.2±2.5	62.7±0.9	54.2±1.6	**67.2±1.3**	64.0±1.0	70.9±3.4	48.0±1.9
128 Examples								
Gotta	80.8±1.7	**60.0±3.6**	**64.9±1.2**	57.4±1.2	**69.8±1.5**	**66.7±1.8**	**78.6±2.1**	**53.3±1.7**
Gotta-random	79.9±1.0	58.6±4.0	64.3±0.9	57.2±0.9	69.8±1.5	66.0±0.7	78.1±2.1	52.5±4.1
Gotta-MTL	77.2±1.9	54.1±1.7	62.4±1.0	53.1±2.1	65.9±1.9	64.1±1.9	76.5±1.3	47.6±2.1
Gotta-what	**80.9±1.4**	57.8±4.0	64.5±0.6	**57.6±0.6**	67.5±0.9	64.7±1.5	77.7±1.9	52.4±2.3

Table 3: Performance of different model variations across all datasets in F1 scores. We also conduct significant tests for Gotta-random. However, Gotta-random does not significantly outperform FewshotQA (p-value $\gg$ 0.05).

Context: *"...Written by Shakira and performed with South African band Freshlyground, the official song of the 2010 FIFA World Cup. Waka Waka (This Time for Africa) expresses the energy and vitality of the African continent. Waka Waka (This Time for Africa) represents what we football fans can expect in South Africa: liveliness, power and dynamic, FIFA president Sepp Blatter said following last week's announcement of the official World Cup song by Fifa and Sony Music ..."* **Question**: *What event was the song "Waka Waka" written for?*		**Context**: *"...The South American Goliath birdeater (Theraphosa blondi) is the world's largest spider, according to Guinness World Records ... They will essentially attack anything that they encounter" Naskrecki said. The spider hunts in leaf litter on the ground at night, so the chances of it encountering a bird are very small, he said. However, if it found a nest, it could easily kill the parents and the chicks, he said, adding that the spider species has also been known to puncture and drink bird eggs"* **Question**: *Goliath is the name for a South American spider that eats what?*	
Answers	**FewshotQA, Gotta-random**: *Football* **Gotta**: *2010 FIFA World Cup* **Ground truth**: *2010 FIFA* or *2010 FIFA World Cup*	**Answers**	**FewshotQA:** Chick; **Gotta-MTL**: *A dog* **Gotta**: *Birds* **Ground truth**: *Birds*

Figure 3: Answers generated by different models for two test cases from TriviaQA. We match the color of the generated answers with their occurrences in the text if they are in the text. In both cases, Gotta successfully generates the correct answer, whereas baselines without entity masking can not accurately recover the entity-level details.

correct answer. In contrast, although Gotta-MTL generates the answer *A dog* that a spider could eat, it is still a wrong answer and does not even appear in the context. This difference perfectly demonstrates that prompt-tuning is beneficial to building connections between entities in the same context. Although FewshotQA returns an answer within the context, the answer is too trivial to answer the question. These two cases provide evidence to validate that entity-aware masking and prompt-style data augmentation in our proposed Gotta are both essential to acquiring the capability of deeply understanding the complicated semantics in questions and contexts.

4.3.5 Effect of Augmented Data As shown in Figure 4, we proceed to study the actual effect of augmented data on the overall performance by investigating the relationship between **average augmented example per training example** and **relative performance improvement**. With the growth of average augmented data per training example, the performance gain is generally larger. Recall that we construct augmented data by raising questions on the entities detected in the context

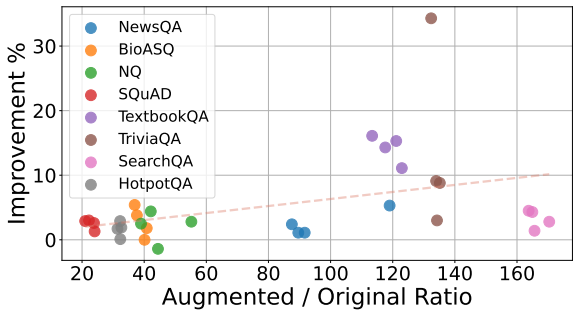

Figure 4: Relative performance improvement w.r.t. the number of augmented data generated per sample. There are in total 32 data points corresponding to each setting on each dataset in Table 2.

of training examples. When there are more entities in the context, GOTTA can learn more about the semantics of the entities and potentially the relations in between, thus having a deeper understanding of the context, thereby further strengthening the QA performance. However, we do observe there is not much gain on SearchQA. Our conjecture is that the contexts of SearchQA are usually very long, so it is rather hard to match the most critical entities. In the extreme case, the entity masking degenerates to MLM, omitting the role of the entities.

5 Conclusion and Future Work

In this work, we propose to incorporate the cloze task to improve neural machine question answering with a few training examples. The key idea is to identify and mask the informative entities in the passage and make the model predict them correctly. Through empirical experimental studies on various QA benchmarks and different few-shot settings, we show that the cloze task indeed benefits the QA task due to its commonalities. We find different ways of incorporating the cloze task improve the QA task while prompt-tuning brings the most. Looking forward, it is of interest to explore QA-dedicated pre-training and ways of pipelining pre-training and prompt-tuning for downstream few-shot QA needs.

Acknowledgements

This paper was partially supported by NSF 1829071, 2106859, 2200274, Cisco and NEC. We thank Ruihan Wu for the useful discussions on the implementation of the framework.

References

[1] A. V. AHO AND M. J. CORASICK, *Efficient string matching: an aid to bibliographic search*, Communications of the ACM, 18 (1975), pp. 333–340.

[2] C. ALBERTI, D. ANDOR, E. PITLER, J. DEVLIN, AND M. COLLINS, *Synthetic qa corpora generation with roundtrip consistency*, in ACL'19, 2019, pp. 6168–6173.

[3] A. ANABY-TAVOR, B. CARMELI, E. GOLDBRAICH, A. KANTOR, G. KOUR, S. SHLOMOV, N. TEPPER, AND N. ZWERDLING, *Do not have enough data? deep learning to the rescue!*, in AAAI'20, 2020, pp. 7383–7390.

[4] J. CAI, Z. ZHANG, F. WU, AND J. WANG, *Deep cognitive reasoning network for multi-hop question answering over knowledge graphs*, in Findings of ACL'21, 2021, pp. 219–229.

[5] R. CHADA AND P. NATARAJAN, *Fewshotqa: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models*, in EMNLP'21, 2021, pp. 6081–6090.

[6] G. CUI, S. HU, N. DING, L. HUANG, AND Z. LIU, *Prototypical verbalizer for prompt-based few-shot tuning*, in ACL'22, 2022, pp. 7014–7024.

[7] J. DEVLIN, M.-W. CHANG, K. LEE, AND K. TOUTANOVA, *Bert: Pre-training of deep bidirectional transformers for language understanding*, in NAACL'19, 2019, pp. 4171–4186.

[8] M. DUNN, L. SAGUN, M. HIGGINS, V. U. GUNEY, V. CIRIK, AND K. CHO, *Searchqa: A new q&a dataset augmented with context from a search engine*, arXiv preprint arXiv:1704.05179, (2017).

[9] A. FISCH, A. TALMOR, R. JIA, M. SEO, E. CHOI, AND D. CHEN, *Mrqa 2019 shared task: Evaluating generalization in reading comprehension*, in Proceedings of the 2nd Workshop on Machine Reading for Question Answering, 2019, pp. 1–13.

[10] T. GAO, A. FISCH, AND D. CHEN, *Making pre-trained language models better few-shot learners*, in ACL'21, 2021, pp. 3816–3830.

[11] K. HAMBARDZUMYAN, H. KHACHATRIAN, AND J. MAY, *Warp: Word-level adversarial reprogramming*, in ACL'21, 2021, pp. 4921–4933.

[12] S. HU, N. DING, H. WANG, Z. LIU, J. WANG, J. LI, W. WU, AND M. SUN, *Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification*, in ACL'22, 2022, pp. 2225–2240.

[13] M. JOSHI, D. CHEN, Y. LIU, D. S. WELD, L. ZETTLEMOYER, AND O. LEVY, *Spanbert: Improving pre-training by representing and predicting spans*, TACL, 8 (2020), pp. 64–77.

[14] M. JOSHI, E. CHOI, D. S. WELD, AND L. ZETTLEMOYER, *Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension*, in ACL'17, 2017, pp. 1601–1611.

[15] A. KEMBHAVI, M. SEO, D. SCHWENK, J. CHOI, A. FARHADI, AND H. HAJISHIRZI, *Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension*, in CVPR'17, 2017, pp. 4999–5007.

[16] D. P. KINGMA AND J. BA, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).

[17] T. KWIATKOWSKI, J. PALOMAKI, O. REDFIELD, M. COLLINS, A. PARIKH, C. ALBERTI, D. EPSTEIN, I. POLOSUKHIN, J. DEVLIN, K. LEE, ET AL., *Natural*

questions: A benchmark for question answering research, TACL, 7 (2019), pp. 453–466.

[18] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, *Albert: A lite bert for self-supervised learning of language representations*, in ICLR'20, 2020.

[19] B. Lester, R. Al-Rfou, and N. Constant, *The power of scale for parameter-efficient prompt tuning*, in EMNLP'21, 2021, pp. 3045–3059.

[20] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension*, in ACL'20, 2020, pp. 7871–7880.

[21] P. Lewis, L. Denoyer, and S. Riedel, *Unsupervised question answering by cloze translation*, in ACL'19, 2019, pp. 4896–4910.

[22] L. Liu, B. Du, J. Xu, Y. Xia, and H. Tong, *Joint knowledge graph completion and question answering*, in KDD'22, 2022, pp. 1098–1108.

[23] X. Liu, Y. Zheng, Z. Du, M. Ding, Y. Qian, Z. Yang, and J. Tang, *Gpt understands, too*, arXiv preprint arXiv:2103.10385, (2021).

[24] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Roberta: A robustly optimized bert pretraining approach*, arXiv preprint arXiv:1907.11692, (2019).

[25] R. Logan IV, I. Balažević, E. Wallace, F. Petroni, S. Singh, and S. Riedel, *Cutting down on prompts and parameters: Simple few-shot learning with language models*, in Findings of ACL'22, 2022, pp. 2824–2835.

[26] S. Min, M. Lewis, H. Hajishirzi, and L. Zettlemoyer, *Noisy channel language model prompting for few-shot text classification*, in ACL'22, 2022, pp. 5316–5330.

[27] R. Puri, R. Spring, M. Shoeybi, M. Patwary, and B. Catanzaro, *Training question answering models from synthetic data*, in EMNLP'20, 2020, pp. 5811–5826.

[28] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, *Squad: 100,000+ questions for machine comprehension of text*, in EMNLP'16, 2016, pp. 2383–2392.

[29] O. Ram, Y. Kirstain, J. Berant, A. Globerson, and O. Levy, *Few-shot question answering by pretraining span selection*, in ACL'21, 2021, pp. 3066–3079.

[30] T. Schick and H. Schütze, *Exploiting cloze-questions for few-shot text classification and natural language inference*, in EACL'21, 2021, pp. 255–269.

[31] ———, *Generating datasets with pretrained language models*, in EMNLP'21, 2021, pp. 6943–6951.

[32] ———, *It's not just size that matters: Small language models are also few-shot learners*, in NAACL'21, 2021, pp. 2339–2352.

[33] D. Tam, R. R. Menon, M. Bansal, S. Srivastava, and C. Raffel, *Improving and simplifying pattern exploiting training*, in EMNLP'21, 2021, pp. 4980–4991.

[34] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman, *Newsqa: A machine comprehension dataset*, in Proceedings of the 2nd Workshop on Representation Learning for NLP, 2017, pp. 191–200.

[35] G. Tsatsaronis, G. Balikas, P. Malakasiotis, I. Partalas, M. Zschunke, M. R. Alvers, D. Weissenborn, A. Krithara, S. Petridis, D. Polychronopoulos, et al., *An overview of the bioasq large-scale biomedical semantic indexing and question answering competition*, BMC bioinformatics, 16 (2015), pp. 1–28.

[36] D. Vrandečić and M. Krötzsch, *Wikidata: a free collaborative knowledgebase*, Communications of the ACM, 57 (2014), pp. 78–85.

[37] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, *Glue: A multi-task benchmark and analysis platform for natural language understanding*, in ICLR'19, 2019.

[38] J. Wang, C. Wang, M. Qiu, Q. Shi, H. Wang, J. Huang, and M. Gao, *Kecp: Knowledge enhanced contrastive prompting for few-shot extractive question answering*, arXiv preprint arXiv:2205.03071, (2022).

[39] Q. Wang, L. Yang, B. Kanagal, S. Sanghai, D. Sivakumar, B. Shu, Z. Yu, and J. Elsas, *Learning to extract attribute value from product via question answering: A multi-task approach*, in KDD'20, 2020, pp. 47–55.

[40] J. Wei and K. Zou, *Eda: Easy data augmentation techniques for boosting performance on text classification tasks*, in EMNLP'19, 2019, pp. 6382–6388.

[41] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le, *Unsupervised data augmentation for consistency training*, in NeurIPS'20, 2020, pp. 6256–6268.

[42] Y. Yang, C. Malaviya, J. Fernandez, S. Swayamdipta, R. Le Bras, J.-P. Wang, C. Bhagavatula, Y. Choi, and D. Downey, *Generative data augmentation for commonsense reasoning*, in Findings of EMNLP'20, 2020, pp. 1008–1025.

[43] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning, *Hotpotqa: A dataset for diverse, explainable multi-hop question answering*, in EMNLP'18, 2018, pp. 2369–2380.

[44] N. Zhang, L. Li, X. Chen, S. Deng, Z. Bi, C. Tan, F. Huang, and H. Chen, *Differentiable prompt makes pre-trained language models better few-shot learners*, in ICLR'21, 2021.

[45] J. Zhao, Z. Guan, and H. Sun, *Riker: Mining rich keyword representations for interpretable product question answering*, in KDD'19, 2019, pp. 1389–1398.

[46] T. Zhao, N. Bian, C. Li, and M. Li, *Topic-level expert modeling in community question answering*, in SDM'13, 2013, pp. 776–784.

[47] Z. Zhong, D. Friedman, and D. Chen, *Factual probing is [mask]: Learning vs. learning to recall*, in NAACL'21, 2021, pp. 5017–5033.