Crafting In-context Examples according to LMs' Parametric Knowledge

Yoonsang Lee[◇,♡,*] **Pranav Atreya**[♠,*] **Xi Ye**[◇] **Eunsol Choi**[◇]
[◇]The University of Texas at Austin, [♡]Seoul National University, [♠]UC Berkeley
lysianthus@snu.ac.kr, pranavatreya@berkeley.edu
xiye@cs.utexas.edu, eunsol@utexas.edu

Abstract

In-context learning has been applied to knowledge-rich tasks such as question answering. In such scenarios, in-context examples are used to trigger a behaviour in the language model: namely, it should surface information stored in its parametric knowledge. We study the construction of in-context example sets, with a focus on the parametric knowledge of the model regarding in-context examples. We identify 'known' examples, where models can correctly answer from its parametric knowledge, and 'unknown' ones. Our experiments show that prompting with 'unknown' examples decreases the performance, potentially as it encourages hallucination rather than searching its parametric knowledge. Constructing an in-context example set that presents both known and unknown information performs the best across diverse settings. We perform analysis on three multi-answer question answering datasets, which allows us to further study answer set ordering strategies based on the LM's knowledge about each answer. Together, our study sheds lights on how to best construct in-context example sets for knowledge-rich tasks.[1]

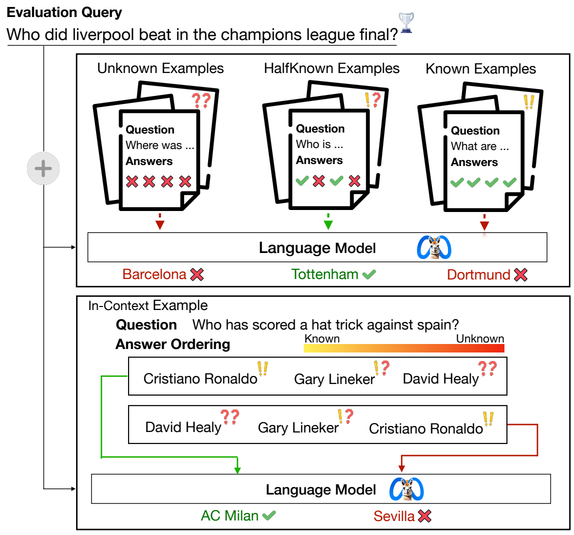

Figure 1: We study how an LM's knowledge of in-context examples impacts their effectiveness. On the top box, we construct three sets of in-context examples, Unknown, HalfKnown, and Known, differing in its difficulty (Section 3). On the bottom box, we construct two in-context examples, which contain the same question and answer set but answers are sorted differently: one in increasing amount of parametric knowledge and one in decreasing difficulty.

1 Introduction

Large language models (LLMs) can perform competitively on knowledge-rich tasks such as question answering via in-context demonstrations (Brown et al., 2020). In such scenarios, in-context examples are used not only to teach the LLM the mapping from inputs to outputs, but also to invoke the LLM's parametric knowledge. Given such role of in-context examples, we examine how the LLM's parametric knowledge of in-context examples impact the effectiveness of in-context examples.

Let's imagine a very challenging in-context example set, where LLMs cannot answer any of in-context examples from its parametric knowledge. For example, in-context examples can query knowledge about recent events that happened after pretraining. These in-context examples will teach the model to generate plausible-looking responses, but may encourage hallucination as a result. On the other hand, if we only provide in-context examples where LLM can easily answer, would LLM learn to make an educated guess on more challenging evaluation examples?

We pose a suite of research questions connecting parametric knowledge of an LM on in-context examples and its impact on model predictions. Figure 1 provides our study overview. We mainly evaluate on multi-answer QA datasets (Min et al., 2020; Malaviya et al., 2023; Amouyal et al., 2022), a challenging knowledge-rich task, and a math QA

[*]Equal Contribution, work was done at UT Austin.
[1]Our code is available at https://github.com/lilys012/known_examples.

dataset (Cobbe et al., 2021), which requires reasoning from LLM. Multi-answer QA datasets further allows a controlled study where we fix the question and vary a choice of answer from a set of valid answers, or how we order answers based on model's parametric knowledge of individual answer.

We first compare providing 'known' or 'unknown' in-context examples (Section 3). We operationalize 'known' in-context examples as those LM can correctly predict with in-context learning. We do not observe a clear winner between two choices, with results varying depending on the dataset. Throughout all datasets, however, providing in-context examples that have a mixture of known and unknown information leads to superior performance compared to solely known or unknown in-context examples.

Our next analysis focuses on the ordering of multi-answer set while fixing in-context example set (Section 4, 5). Compared to randomly ordering valid answers, semantically meaningful ordering brings substantial changes in model predictions. Even alphabetical ordering of answer set changes predicted answers substantially, prompting model to generate 1.5 more answer on average than when shown randomly sorted answer set. We further find that ordering the answer set of in-context examples in descending order of model knowledge often leads to performance gains. Together, our work suggests best practices for crafting in-context examples, with relation to its parametric knowledge, for knowledge-intensive tasks.

2 Experimental Settings

We first describe our evaluation setting which centers around multi-answer QA datasets.

2.1 Dataset

We evaluate on three multi-answer QA datasets: (1) AmbigQA (Min et al., 2020) contains a subset of questions from the Natural Questions (Kwiatkowski et al., 2019) dataset, namely those marked as ambiguous in the sense that depending on the interpretation they can have multiple correct answers. (2) QAMPARI (Amouyal et al., 2022) consists of questions whose set of correct answers necessarily span multiple paragraphs in the document from which they were retrieved. The dataset was originally developed to evaluating retrieval methods, and we repurpose it to create a challenging closed-book QA setting. (3) QUEST (Malaviya et al., 2023) dataset is constructed by formulating queries that define implicit set operations over Wikipedia entities. We report data the dataset statistics in Appendix A.

2.2 Evaluation Metrics

Given a question q, the model will predict a set of answers $\hat{a} = \{a_1, a_2, ..., a_m\}$, where each $a_i = (w_{i_1}, w_{i_2}, ..., w_{i_{|a_i|}})$. is a sequence of tokens for a single answer. We denote $a^* = \{a_1^*, a_2^*, ..., a_n^*\}$ as the ground truth answers to the same question.

We use standard token match metrics for evaluating answer accuracy, Exact Match (EM) and F1-score (Joshi et al., 2017). EM assigns a score of 1 if the predicted answer equals to the ground truth answer, while F1-score is calculated over the tokens in the answer. We use metrics for multi-answers introduced in prior work (Min et al., 2020), which we describe below for completeness.

Answer-level Exact Match ($\mathbf{F1_{EM}}$) As predicting the exact ground truth answer set correctly is very challenging, we report the F1-score of answer-level exact match, denoted as $F1_{EM}$. For an answer a and reference answers set S, we define a correctness score $c(f, a, S) = f(a, S)$ with respect to function f. We use $f(a, S) = \mathbb{1}(a \in S)$ here. Then, we calculate the F1-score over set-level precision and recall according to c.

$$P = \frac{\sum_{i=1}^{m} c(f, a_i, a^*)}{m}, R = \frac{\sum_{j=1}^{n} c(f, a_j^*, \hat{a})}{n}$$

$$F1_{EM} = \frac{2 \times P \times R}{P + R}$$

Answer-level F1 ($\mathbf{F1_{F1}}$) The generated answer may be semantically equivalent to one of the ground truth answers, without being lexically equivalent (e.g., "Friends" and "The TV show Friends"). To account for such semantic equivalences, we use $F1$ score between the tokens of two answer strings instead of the exact match as a correctness score, $f(a, S) = max_{a' \in S}(F1(a, a'))$. Then, we compute F1-score over set-level precision and recall as above.

Statistical Testing As our evaluation datasets are relatively small, we conduct paired bootstrap tests throughout most of our experiments, highlighting results that outperform baseline with p value of ≤ 0.05.

2.3 Base Models

Language Model We use Llama-2 (Touvron et al., 2023) language model (13B). Our inital pilot showed the base model achieves better in-context learning performance on our tasks, whereas the chat variants may produce lengthy and unnecessary text regarding the answers under the same prompt (e.g., chat series can generate "Sure, I'd be happy to help! Here are the answers to your questions:").

In-context Example Retriever Prior work (Rubin et al., 2021) has established that using semantically similar in-context examples improves the performance of in-context learning significantly. Throughout our study, we often retrieve top 5 most similar in-context examples from the entire training set for each dataset to form the prompt. We place in-context examples in decreasing order of similarity, such that the most similar example will be presented closest to the evaluation question. We measure example similarities by encoding each question with a SimCSE sentence embedding model (Gao et al., 2021) and computing their dot product.

3 Known Examples vs. Unknown Examples

Prior work has studied a few characteristics of successful in-context example set, such as label distribution in the in-context example set (Min et al., 2022). We evaluate in-context examples with respect to model's parametric knowledge, whether an "known" or "unknown" in-context example is better. We operationalize "known" ones as the ones where LLMs can get the answers correctly from its own parametric knowledge, and "unknown" ones as those that cannot be answerable from its parametric knowledge.

3.1 Single Answer Study

In this study, we provide only one answer from multi answer set in in-context examples. For example, if question in in-context example is "who was the president of U.S.?", we can either provide a famous president or a lesser-known president as an answer. Both are "correct" answers, but which answer would lead to better model performance?

For each question in our evaluation set, we will retrieve top five most similar examples in training set as in-context examples. Below we discuss how to choose a single gold answer from gold answer set for each selected in-context examples. We will measure perplexity of each answer to approximate how well LM 'knows' the answer. For each example, a pair of question q and gold answer set $\{a_1^*, a_2^*, ...a_n^*\}$, we form a prefix p by prepending top five most similar examples to the query q.[2] Then, we compute the length normalized perplexity of each answer a_i^* and prefix p as follows:

$$PP(a_i^*|p) = \prod_{j=1}^{|a_i^*|} P(w_{ij}|p, w_{i1}, ..., w_{i(j-1)})^{-\frac{1}{|a_i^*|}}$$

We will order the gold answer set in descending order of perplexity, and select an answer at the x-th quantile. This way, an answer at the 100% quantile represents the most 'known' answer, as its perplexity is the lowest among the gold answers.

Figure 2 presents the $F1_{EM}$ score among various x-th quantile. We observe a clear trend across all three datasets, that using a 'known' answer leads LM to generate more accurate answer. However, these in-context examples are incomplete, only presenting one answer while there are multiple valid answers. This leads to low performance overall, as LM will only generate a single answer. In the next section, we formulate fixed in-context example set where gold answer set is preserved.

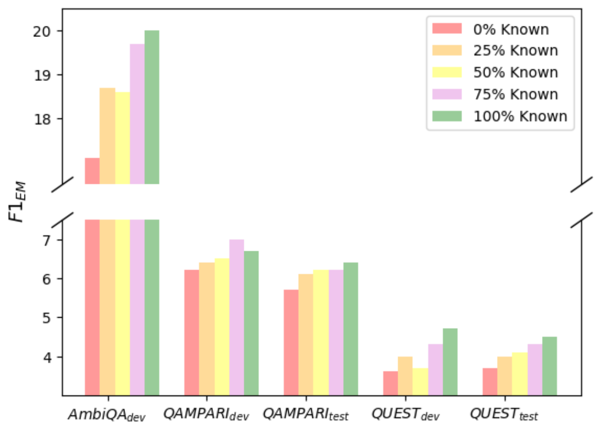

Figure 2: Results of single answer study. Only an answer at the x-th quantile of perplexities in decreasing order is presented in each in-context example. As the model gets exposed to more known answers, the performance tend to increase.

3.2 In-context Example Set Study

Multi answer QA dataset We create four sets of in-context examples, differing in its difficulty for a given LM.

[2] We present an example prefix in Appendix F.

	AmbigQA		QAMPARI		QUEST	
	$F1_{EM}$	$F1_{F1}$	$F1_{EM}$	$F1_{F1}$	$F1_{EM}$	$F1_{F1}$
Random	18.0	28.9	10.3 / 10.0	20.8 / 19.3	4.0 / 3.4	12.1 / 11.0
Unknown	17.2*	28.2*	10.9*/ 10.6	22.0*/ 20.2*	4.4*/ 3.7*	**13.2***/ 11.9*
HalfKnown	**18.5***	**29.5***	**11.3***/ **11.2***	**22.6 / 20.9***	**4.9***/ **4.0***	13.1*/ 11.9*
Known	18.3*	29.0*	9.8 / 9.9	19.7 / 18.6	4.3*/ 3.9*	12.8*/ **12.0***

Table 1: Results comparing known example and unknown example. Using half-known example outperforms other settings. For QAMPARI and QUEST datasets, we present development set performance and then test set performance in each cell. We put * on scores that are significantly different from that of Random in-context examples set, and bold the highest performing set for each metric.

- **UNKNOWN**: examples for which the LM possesses no knowledge of the answers. Operationally, these are examples when LM is prompted with five most similar examples, LM will predict zero answer correctly (i.e. zero $\mathbf{F1_{EM}}$ score).
- **RANDOM**: randomly sampled examples. Since LM possesses no knowledge to majority of the examples, these exhibit 0.18 $\mathbf{F1_{EM}}$ score on average.
- **HALFKNOWN**: examples for which the LM possesses roughly half knowledge of the answers (i.e. 0.5 $\mathbf{F1_{EM}}$ score).
- **KNOWN**: examples for which the LM possesses full knowledge of the answers (i.e. 1.0 $\mathbf{F1_{EM}}$ score).

As prior work (Rubin et al., 2021) has established that the similarity of in-context example to the query correlates strongly with the model's performance, we control for this confounding factor. We compute the average similarity for each in-context example candidate to other in-context example candidates in the candidate set (training set). Then, we choose a fixed number of in-context examples whose average similarity value is close to the median value.[3] From this candidate set, we sample five examples for each condition and use it as a fixed in-context examples across all questions in the evaluation dataset. To further reduce randomness, we sample multiple sets of five example set for each condition and report the average performance (by default, four sets are sampled and two sets are sampled for HALFKNOWN and KNOWN set in QUEST because of lack of examples with sufficient model knowledge).

[3]We choose 999 examples for AmbigQA and QAMPARI, and 499 for QUEST (as QUEST only has 1251 training examples), half from below median, half from above median. For QUEST, we could not find enough examples with where model score full $\mathbf{F1_{EM}}$ score, so we selected highest scoring examples. The mid-range is (0.245, 0.264), (0.294, 0.296), (0.326, 0.373) for AmbigQA, QAMPARI, and QUEST.

Unknown	Random	HalfKnown	Known
33.1	34.8	**36.4**	32.0

Table 2: The accuracy on GSM8K dataset. Accuracy is expressed as the percentage of correct answers over the entire test dataset, which consists of 1319 queries.

Table 1 presents the performance of each in-context example set on three datasets. On AmbigQA and QAMPARI dataset, we observe HALFKNOWN in-context example set outperforms all other settings. On QUEST dataset, the trend was less clear because of the small size of evaluation and training dataset. We hypothesize HALFKNOWN encompasses in-context examples that contain both answers that the model knows and doesn't know. This may successfully prompt LMs to leverage parametric knowledge and to make educated guesses.

Single Answer Math QA Dataset In this section, we explore constructing in-context example sets with varying "knownness" for single-answer QA task. We chose GSM8K (Cobbe et al., 2021) dataset, a commonly used dataset for investigating the reasoning capabilities of LLMs. GSM8K consists of 8,500 natural language questions requiring arithmetic reasoning for obtaining an answer. To evaluate parametric knowledge available to solve each training example with the LM, we prompt each example with the 8-shot example set taken from Wei et al. (2022b) and classified as correct, wrong, or invalid, where invalid indicates that the model did not produce an answer. We construct four in-context example sets:

- UNKNOWN set includes randomly selected six examples that model answered incorrectly.
- RANDOM set includes randomly selected six ex-

amples from entire training dataset. The LM correctly answer questions in training set for 20% of questions.

- HALFKNOWN set includes three correct and three wrong examples.
- KNOWN set includes randomly selected six examples that model answered correctly.

We select six examples four times and report the averaged accuracy in Table 2. *HalfKnown* set achieves the highest accuracy, repeating the trend from multi-answer QA datasets.

4 Ordering Answers Based on LM's Knowledge

Prior work suggests that the ordering of in-context examples significantly impacts performance, with more relevant examples being most beneficial when placed last (Zhao et al., 2021). Yet, no prior work has studied the ordering of answers inside each in-context example. We investigate this here. Following our previous study, our focus is on **parametric** knowledge of LMs being prompted. Specifically, we question whether placing answers based on how well the model knows about answers improves the performance. We first introduce various ordering strategies for the gold answer set.

4.1 Ordering Strategies

We present strategies to order the answer set of each example, a pair of question q and its gold answer set $a^* = \{a_1^*, a_2^*, ..., a_n^*\}$, which will be used as an in-context example.[4] We present two baselines and two methods (PERPLEXITY, GREEDY) for ordering the gold answer set of each in-context example based on model's parametric knowlege.

Baselines The RANDOM baseline randomly orders answers, and ALPHABET orders answers alphabetically. While alphabetical ordering is not relevant to model's parametric knowledge of the answer, prior work (Madaan et al., 2022) has shown that consistent ordering of labels can improve the performance of fine-tuned LLM's predictions.

Knowledge-Aware Ordering We decide ordering based on the **perplexity** of individual answer given the prefix, or by performing greedy **constrained decoding** given the prefix. We use the same prefix as in Section 3.1, a concatenation of five in-context examples. Each ordering strategy will yield two orderings of answers, which either sorts the answers in the descending order of model's parametric knowledge or ascending order (denoted as REVERSE).

Input: LM $\mathcal{M}$, Prefix p, Gold answer set $a^* = \{a_1^*, \ldots, a_n^*\}$, where each gold answer is a token sequence (i.e., $a_i^* = (w_{i_1}, \ldots w_{i_{|a_i^*|}})$)
Output: Ordered answer indices of the gold answer set

$I_1 \leftarrow \{w_{1_1}, ..., w_{n_1}\}$
$u \leftarrow 1$
while $I_1 \neq \emptyset$ **do**
  $t \leftarrow 0$
  repeat
    $t \leftarrow t + 1$
    $o_t \leftarrow \mathbf{argmax}_{w \in I_t} P_{\mathcal{M}}(w|p)$
    $p \leftarrow [p; o_t]$
    $I_{t+1} \leftarrow \{w_{i_{t+1}} | w_{i_t} == o_t\}$
  until $\exists a_{k_u}^* == (o_1, \ldots, o_t)$ {this assigns k_u the index of completed answer}
  $I_1 \leftarrow I_1 \setminus \{w_{k_u 1}\}$
  $u \leftarrow u + 1$
return $\{k_1, \ldots, k_n\}$

Figure 3: Algorithm for constrained decoding for GREEDY ordering.

- PERPLEXITY: We compute the length normalized perplexity of each answer a_i^* and prefix p as used in Section 3.1. Then, we sort the answers in ascending order of these perplexities, resulting in 'known' answers placed earlier.
- GREEDY: We arrange the gold answers $a^* = \{a_1^*, a_2^*, ..., a_n^*\}$ by performing a beam search decoding in a greedy manner, constrained to permissible tokens. There will be two loops, outer loop for selecting the first token of the generated answer, and inner loop for completing the chosen first token.
  Figure 3 presents the pesudocode, which we explain below. Let's denote a_i^* as a sequence of tokens $(w_{i_1}, w_{i_2}, ..., w_{i_{|a_i^*|}})$ for the i-th answer. At each decoding step t, a set of permissible tokens I_t is constructed. Initially, $I_1 = \{w_{1_1}, w_{2_1}, ..., w_{n_1}\}$, a set of the first token for each potential answer. We choose a token from this set that has the highest likelihood given the prompt, i.e., $o_1 = \text{argmax}_{w \in I_1} P(w|p)$. Then, we update the prefix $p \leftarrow [p; o_1]$. This initiates the inner loop, setting $I_2 = \{w_{i_2} | w_{i_1} == o_1\}$ as a set of second token of answers who starts with the selected first token. This continues until one of the answers a_{k_1} is fully generated. Af-

[4]As reordering process is computationally expensive, proportional to the number of answers, we only consider examples that have less than 20 answers. This results in exclusion of 1 example in AmbigQA, 8094 examples in QAMPARI, and none in QUEST.

		S				
		GREEDY	REVERSE GREEDY	PERPLEXITY	REVERSE PERPLEXITY	ALPHABET
$\mathcal{D}_e$	AmbigQA$_{\text{dev}}$	71.7 / 66.0	39.2 / 37.2	69.5 / 65.8	38.1 / 34.2	87.4 / 55.6
	QAMPARI$_{\text{dev}}$	69.6 / 60.0	42.2 / 41.0	58.1 / 54.1	46.3 / 45.9	95.0 / 58.9
	QAMPARI$_{\text{test}}$	70.0 / 65.7	43.0 / 41.7	58.8 / 55.8	45.0 / 44.2	94.9 / 58.1
	QUEST$_{\text{dev}}$	78.4 / 63.9	47.2 / 45.8	57.1 / 51.5	49.3 / 48.5	95.7 / 52.1
	QUEST$_{\text{test}}$	81.0 / 63.3	45.7 / 45.3	57.6 / 52.5	48.8 / 47.5	95.6 / 50.8
	Average	74.1	43.5	60.2	45.5	**93.7**

Table 3: Percentage of generated answer ordering matching in-context examples answer ordering. In each cell, we present the percentage from using corresponding answer ordering strategy first ($\phi(S, \mathcal{D}_t^S, \mathcal{D}_e, \mathcal{M})$) and the percentage for randomly ordering answers for control ($\phi(S, \mathcal{D}_t^{S_{\text{random}}}, \mathcal{D}_e, \mathcal{M})$).

terwards, an we come back to the outer loop, the initial set of permissible tokens is set to be $I_1 = \{w_{1_1}, w_{2_1}, ..., w_{n_1}\} \setminus \{w_{t_1}\}$ excluding $a_{k_1}^*$ which has been already generated. This process continues until all answers has been generated, with a time complexity of O($n|a_i^*|$).

5 Results for Answer Ordering Strategies

Having introduced strategies for ordering answers for in-context examples, we study how this impacts the generation of answers. We first evaluate whether the generated answers mimic the ordering of answers in in-context examples. Then, we evaluate whether the ordering impacts the size and the accuracy of predicted answer set.

5.1 Does the predicted answer set follow the ordering of in-context answer set?

Throughout in-context learning, the model is expected to learn the pattern shown in the demonstrations. We assess the generated answers to observe whether the model has followed the particular ordering shown in the in-context examples.

Metric We introduce a metric $\phi(S, \mathcal{D}_t^{S^t}, \mathcal{D}_e, \mathcal{M})$. This measures how much LM $\mathcal{M}$ follows the answer ordering strategy S on evaluation dataset $\mathcal{D}_e$ when using in-context examples from training dataset $\mathcal{D}_t$ whose answered are ordered according to S^t.[5] When S matches S^t, this metric will measure how much predicted outputs mimic the answer ordering strategy of in-context examples.

Let's denote $\hat{a}_i = \{a_{i_1}, a_{i_2}, ..., a_{i_m}\}$ be the list of predicted m answers for the i-th example of an evaluation dataset $\mathcal{D}_e$, following its generation order from model M. We reorder the predicted answers from $\hat{a}_i$ with respect to S and denote $f(a_{ij})$ to be the index of a_{ij} in the newly ordered set.

[5]We assume retrieving five most similar in-context examples for each evaluation example throughout this study.

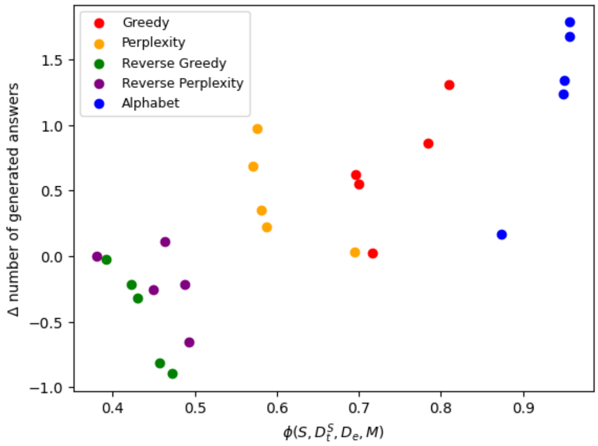

Figure 4: $\phi(S, \mathcal{D}_t^S, \mathcal{D}_e, \mathcal{M})$ vs. the number of generated answers across three datasets. Instead of the raw number of answer set, we report the size difference compared to the answer set generated from random ordering. As ϕ increases, which signifies how faithfully LM follows the ordering strategy in in-context examples, the model generates more answers.

For each consecutive answer pair in $\hat{a}_i$, we evaluate whether their order is preserved after reordering. Then we count the number of consecutive answer pairs that have preserved the ordering, which is $P_i = \sum_{j=1}^{m-1} \mathbb{1}(f(a_{ij}) < f(a_{i(j+1)})$. Similarly, $N_i = \sum_{j=1}^{m-1} \mathbb{1}(f(a_{ij}) > f(a_{i(j+1)})$ represents the number of pairs that violates the ordering. Then, we compute micro average over $\mathcal{D}_e$.

$$\phi(S, \mathcal{D}_t^{S^t}, \mathcal{D}_e, \mathcal{M}) = \frac{100 \cdot \sum_{i \in D_e} P_i}{\sum_{i \in D_e} (P_i + N_i)}$$

Results The results are presented in Table 3. For each $\phi(S, \mathcal{D}_t^S, \mathcal{D}_e, \mathcal{M})$, we also report $\phi(S, \mathcal{D}_t^{S_{\text{random}}}, \mathcal{D}_e, \mathcal{M})$ as a control. We found that in every cell, the first number is higher than the second number, suggesting that the model follows to the answer ordering pattern presented in the in-context examples. We found this is particularly

QAMPARI	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$
RANDOM	26.3 / 25.2	11.7 / 10.9	13.8 / 12.9	25.3 / 22.4
GREEDY	26.4 / 25.7	12.2 / **11.9***	14.2 / **14.0***	25.6 / 22.6
PERPLEXITY	26.7 / 26.4*	12.4* / 11.6*	**14.6*** / 13.9*	**25.8** / **22.9**
REVERSE GREEDY	26.5 / 25.8	11.6 / 10.1*	13.9 / 12.4	25.1 / 21.8
REVERSE PERPLEXITY	**27.0** / **26.7***	11.7 / 11.0	14.0 / 13.3	25.2 / 22.5
ALPHABET	24.5* / 23.5*	**12.7*** / 11.8*	14.3 / 13.6	24.7 / 22.6
QUEST	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$
RANDOM	23.9 / **24.8**	17.9 / 19.7	18.3 / 19.9	27.2 / 27.8
GREEDY	23.8 / **24.8**	**19.6*** / **20.8***	**19.5*** / **20.6***	**28.6*** / **28.4***
PERPLEXITY	**24.3** / **24.8**	19.3* / **20.8***	19.4 / **20.6***	28.0 / **28.4***
REVERSE GREEDY	22.9 / 24.5	17.0 / 18.4*	17.4 / 18.8*	26.3 / 26.5*
REVERSE PERPLEXITY	23.7 / 24.5	17.3 / 19.4	17.7 / 19.4	26.4 / 27.1*
ALPHABET	20.5* / 23.8*	17.6 / 20.4*	17.0 / 20.0	25.0* / 27.0*

Table 4: QA performance for answer ordering strategies on Llama2 (13B) model. P_{EM} and R_{EM} are precision and recall for calculating $F1_{EM}$. We present development set performance and then test set performance in each cell. Blue color indicates improved performance compared to Random and red indicates the opposite. We put * on scores that are significantly different from that of Random ordering.

true for ALPHABET ordering, which is probably the easiest pattern to learn.

We further observe that the model is decoding answers such that it will present **confident** answer first (following the orders of GREEDY and PERPLEXITY), even when answers in in-context example is randomly ordered. Even after introducing consistent ordering (presenting less confident answer first), the model shows propensity to present confident answer first (values for REVERSE GREEDY and REVERSE PERPLEXITY are below chance (50) consistently).

5.2 Does ordering impact the number of generated answers?

Unlike in simpler QA tasks where there is exactly one gold answer, models have to decide how many answers to generate. Would consistent ordering of answers allow the model to generate more answers?

We report the number of generated answers for each ordering strategy in Figure 4. We find that generation order impacts the number of generated answer, with ALPHABET ordering substantially increasing the number of generated answers the most. The results further suggest that an ordering pattern that is easier for the model to learn can prompt LM to generate more answers.

5.3 Does the ordering impact the QA performance?

Lastly, we examine the end task (QA) performance of different answer ordering strategies. Table 4 presents the results on QAMPARI and QUEST datasets. Overall, we see that answer ordering does not bring large impact in final end task performance, but notice consistent patterns. Overall, presenting more confident answers first (GREEDY and PERPLEXITY) yielded better results than their REVERSE counterparts. GREEDY and PERPLEXITY show gains mostly in recall, leading to increase in both $F1_{EM}$ and $F1_{F1}$. Arbitrary, yet consistent ordering such as ALPHABET does not improve model performance, sometimes rather leading to lower performance. Our results suggests that ordering 'known' answer first in in-context examples can improve model performance.

For AmbigQA dataset, we do not see any statistically significant results compared to random ordering. We think this might be because of smaller average answer set size of AmbigQA dataset compared to that of other datasets (2-3 vs. 10+ answers).

6 Discussions and Limitations

Varying Base LMs Throughout our experiments, we use Llama2 13B model as our LM. Would our conclusions hold for other LMs? We conduct a few experiments on OPT 13B (Zhang et al., 2022) to investigate this. With respect to following the ordering strategy of in-context examples (Section 5.1, 5.2), we find that the results hold for OPT LLM model as well. However, the end task performance results are somewhat mixed. Overall, we find that ordering between answers will bring significant differences in end task performance **only** when model exhibit sufficient parametric knowledge of subset of answers, and the number of answers getting reordered are larger.

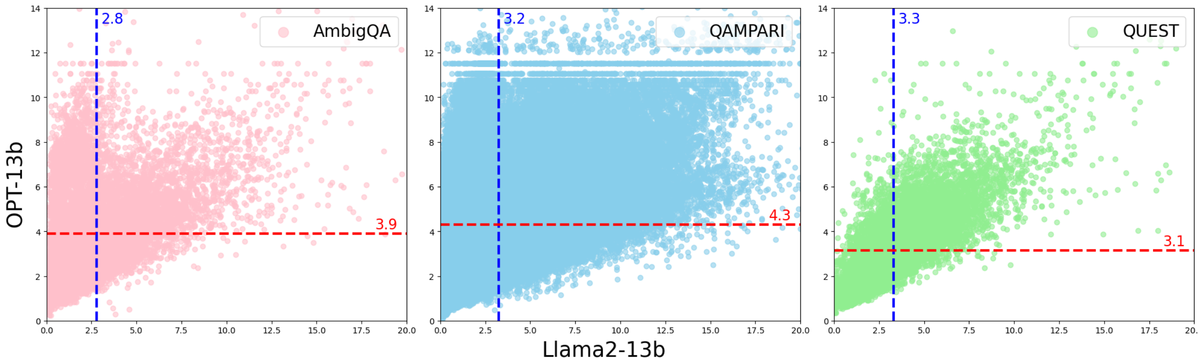

Figure 5: Plots of log answer perplexities from Llama2-13b (x-axis) and OPT-13b (y-axis). Horizontal and vertical lines indicate the mean value of log perplexities with respect to each LM. In all datasets, Llama2 outperforms OPT in its parametric knowledge, and the answers mostly report higher perplexity with OPT compared to Llama2.

QUEST	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$	# pred ans
RANDOM	14.6 / 18.4	11.6 / 16.1	12.0 / 15.6	21.3 / 23.8	7.56 / 7.78
GREEDY	15.7 / **18.6**	**16.6*** / **18.0***	**14.9*** / **17.0***	**23.7*** / **25.2***	9.93 / 9.87
PERPLEXITY	16.1 / 18.3	14.8* / 17.0*	13.9* / 16.2	22.6 / 24.5	8.42 / 8.85
REVERSE GREEDY	14.5 / 17.4*	10.7 / 13.8*	10.2 / 13.8*	19.6 / 22.1*	6.55 / 6.67
REVERSE PERPLEXITY	15.0 / 17.9	14.3* / 15.4	13.2 / 15.1	22.4 / 23.5	7.41 / 7.44
ALPHABET	**16.3*** / 17.6*	15.9* / 17.3*	14.7* / 16.3*	23.0* / 24.1*	**10.12** / **10.11**

Table 5: QA performance on QUEST for answer ordering strategies with OPT (13B) model. The table is formatted the same as Table 4. In addition, we add a last column (# pred ans) which reports the number of predicted answers.

To measure the amount of parametric knowledge for each dataset for each LM, we plot the perplexity of individual answer in train examples with respect to two language models in Figure 5. Two models exhibit similar knowledge for QUEST, however OPT shows a wider range of perplexities, especially for answers that have low perplexity on Llama2. OPT does not seem to have similar level of parametric knowledge for QAMPARI or AmbigQA datasets, resulting in substantially higher average perplexity number. We observe consistent results of end task performance on QUEST dataset but the results are mostly random on AmbigQA and QAMPARI dataset (see Table 5, full results reported in Table 9 in the appendix). When the model is not familiar enough with the gold answers in in-context examples, knowledge-aware answer ordering might have limited effectiveness.

Random vs. Similar In-Context Examples Prior works have highlighted the importance of relevant in-context examples, such as those based on similarity (Liu et al., 2021) and diversity (Levy et al., 2022). Yet, many studies do not do example specific retrieval and use random examples for its simplicity. Throughout our experiments (except for Section 3.2 which constructs universal in-context set for all examples in evaluation dataset), we retrieved similar in-context examples for each evaluation example. How would our results hold if we use randomly select in-context examples?

We summarize the results here, and report the full results in Appendix E. First, with randomly retrieved in-context examples, models still learn to follow the answer ordering strategy shown in in-context examples but substantially less than when using similar incontext examples. Second, we find that the number of generated answer is affected similarly, with using ALPHABET ordering leads to the highest number of generated answers. However, we see invariant performances on end tasks. Carefully constructing relevant in-context examples is more meaningful than doing it for random in-context examples. This suggests that if you do not have large enough training examples to recover semantically relevant in-context examples, careful construction of prompt might not yield changes in end task performance.

7 Related Work

Multi Label Ordering While not studied extensively under the in-context-learning setting, a recent work (Madaan et al., 2022) studies set generation problem from encoder-decoder model, showing that imposing informative ordering over the label space improves model performance.

Analysis on In-context Learning Many prior works investigate factors that determine the performance of in-context learning (ICL) (Brown et al., 2020), such as the composition of the pretraining dataset (Xie et al., 2022), size of a language model (Wei et al., 2022a), number of pretraining tokens (Touvron et al., 2023), and specific fine-tuning strategy employed (Wei et al., 2021). More closely related to ours, one line of work particularly focuses factors related to the in-context examples, including the choice of verbalizer and templates (Min et al., 2022), order of examples (Lu et al., 2022), and the choice of in-context examples (Liu et al., 2021; Rubin et al., 2021; Ye et al., 2023). While past work is mainly centered around classification tasks, our work studies the task of multi-answer QA, with a focus on how LM's parametric knowledge on in-context examples impact the performance. In particular, our findings suggest using answers with lower perplexity in in-context examples leads to more accurate answer, which is congruent with recent work that shows using lower plexity prompts improves model perplexity in general (Ye and Durrett, 2023; Gonen et al., 2022).

Multi-answer QA Real-world questions could naturally have multiple answers when a question is ambiguous (Min et al., 2020; Stelmakh et al., 2022), when a question is evaluated under different temporal or geographical contexts (Zhang and Choi, 2021), or when a question expects a set of answers (Amouyal et al., 2022; Malaviya et al., 2023). While most prior work tackles multi-answer QA in the open-book setting by retrieving from external corpus (Shao and Huang, 2022; Sun et al., 2023), we study the problem in the close-book setting, which prompts LLMs to generate the answers based on their parametric knowledge.

8 Conclusion

This work presents comprehensive studies on knowledge-aware prompt design for multi-answer QA tasks. Our findings underscore the benefits of having in-context examples that language model is familiar with. First, the HALFKNOWN set, which contains in-context examples where the answers are partially known, aids the model in effectively accessing its parametric knowledge. Second, employing knowledge-aware ordering of presenting answers in descending order based on the model's knowledge enhances the overall process of answer generation. We hope for our findings to offer potential insights into the in-context learning for knowledge intensive tasks.

Limitations

Our study is conducted on open-sourced models, mainly focusing on Llama2. Future work can explore how the results will generalize to larger models. Also, the analysis can be extended to a wide range of tasks that requires different types of reasoning ability.

Acknowledgment

We thank the members of UT Austin NLP community for valuable feedback, especially Thom Lake for suggesting an ablation study and Greg Durrett for providing feedback on the draft. This work is partially funded by a gift from Home Depot.

References

Samuel Joseph Amouyal, Ohad Rubin, Ori Yoran, Tomer Wolfson, Jonathan Herzig, and Jonathan Berant. 2022. Qampari: An open-domain question answering benchmark for questions with many answers from multiple paragraphs. *ArXiv, abs/2205.12665*.

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.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *ArXiv*, abs/2104.08821.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A. Smith, and Luke Zettlemoyer. 2022. Demystifying prompts in language models via perplexity estimation. *ArXiv*, abs/2212.04037.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *ArXiv*, abs/1705.03551.

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.

Itay Levy, Ben Bogin, and Jonathan Berant. 2022. Diverse demonstrations improve in-context compositional generalization. *ArXiv*, abs/2212.06800.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2022. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Aman Madaan, Dheeraj Rajagopal, Niket Tandon, Yiming Yang, and Antoine Bosselut. 2022. Conditional set generation using seq2seq models. In *Conference on Empirical Methods in Natural Language Processing*.

Chaitanya Malaviya, Peter Shaw, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2023. Quest: A retrieval dataset of entity-seeking queries with implicit set operations. *arXiv preprint arXiv:2305.11694*.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*.

Ohad Rubin, Jonathan Herzig, and Jonathan Berant. 2021. Learning to retrieve prompts for in-context learning. *arXiv preprint arXiv:2112.08633*.

Zhihong Shao and Minlie Huang. 2022. Answering open-domain multi-answer questions via a recall-then-verify framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1825–1838, Dublin, Ireland. Association for Computational Linguistics.

Ivan Stelmakh, Yi Luan, Bhuwan Dhingra, and Ming-Wei Chang. 2022. ASQA: Factoid questions meet long-form answers. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8273–8288, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Weiwei Sun, Hengyi Cai, Hongshen Chen, Pengjie Ren, Zhumin Chen, Maarten de Rijke, and Zhaochun Ren. 2023. Answering ambiguous questions via iterative prompting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7669–7683, Toronto, Canada. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Jiacheng Ye, Zhiyong Wu, Jiangtao Feng, Tao Yu, and Lingpeng Kong. 2023. Compositional exemplars for in-context learning. *arXiv preprint arXiv:2302.05698*.

Xi Ye and Greg Durrett. 2023. Explanation selection using unlabeled data for in-context learning. In *Proceedings of EMNLP*.

Michael Zhang and Eunsol Choi. 2021. SituatedQA: Incorporating extra-linguistic contexts into QA. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7371–7387, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

Tony Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*.

A Dataset Statistics

We report the dataset statistics in Table 6.

B Similarity of In-Context Examples

We calculate the similarity score of two in-context examples using SimCSE embeddings of each query. Figure 6 illustrates the similarity distributions across three datasets.

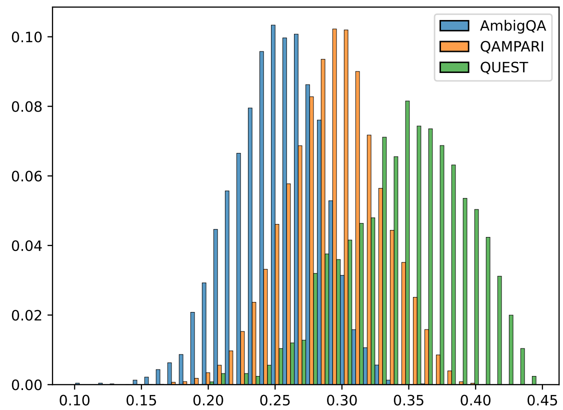

Figure 6: Similarity distributions among in-context example candidates. The x-axis denotes embedding similarity (with SimCSE (Gao et al., 2021) encoder) and the y-axis indicates the percentage of each bin. The median value for each dataset is 0.254, 0.295, 0.350.

C Answer Ordering Strategies

C.1 Single Answer Study

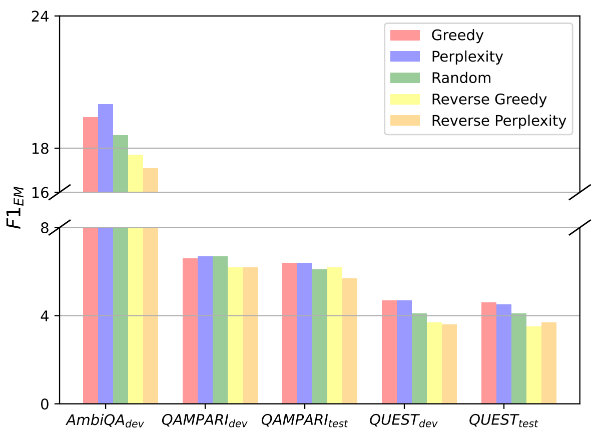

Figure 7: Answer-level Exact Match ($F1_{EM}$) score for demonstrating only one frontmost answer of an ordering methodology.

We examine the effectiveness of answer ordering strategies discussed at Section 4.1. We provide only one answer at the forefront of each ordered answers in in-context examples. Since an answer from GREEDY and PERPLEXITY is 'known' to the model, they may serve as an upper bound of 'known' answer, while REVERSE GREEDY and REVERSE PERPLEXITY may serve as a lower bound. RANDOM exists somewhere between these. The disparities among these are clear, as shown in Figure 7. The results suggest that the model is able to differentiate ordering strategies.

C.2 AmbigQA results

We present the performance of answer ordering strategies on AmbigQA dataset in Table 7.

D Model Variants

We present the results of experiments in Section 5 with OPT 13B model. Table 8 suggests that the model is also capable at capturing the answer ordering patterns. End task performances with OPT 13B over AmbigQA and QAMPARI are presented in Table 9. Consistent answer ordering that places known answers at the front generates more answers, however resulting significant performance gain only in QUEST. We attribute this to the familiarity of LM to the specific task, as OPT shows less familiarity to AmbigQA and QAMPARI when compared to Llama2 (Figure 5).

E Random Examples

We report the results of utilizing random in-context examples in Table 10 and 11.

F Prompts

Throughout Table 12 to Table 15, we present the prompts used in our experiments.

	AmbigQA		QAMPARI			QUEST		
	Train	Dev.	Train	Dev.	Test	Train	Dev.	Test
# Examples	4,615	1,048	50,372	1,000	1,000	1,251	316	1,669
Avg. # of answers	2.8	3.1	14.0	13.2	13.1	10.9	10.7	10.7
Query length	46.9	46.7	67.8	57.7	55.8	54.0	52.2	53.3
Answer length	15.9	14.5	14.4	17.3	16.6	17.2	16.7	17.0
Answer sequence length	45.2	45.4	200.9	228.5	217.6	187.0	179.0	182.4
# Unique answers	10,684	2,999	455,469	12,462	12,464	10,160	3,050	12,367

Table 6: Dataset statistics. Lengths of query, answer, and answer sequence are measured by the length of each string. # Unique answers counts unique answers within each split. Duplicated questions are removed from training sets.

AmbigQA	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$
RANDOM	27.1	17.9	20.0	31.3
GREEDY	27.2	**18.5**	**20.5**	31.7
PERPLEXITY	**27.4**	18.4	**20.5**	**31.8**
REVERSE GREEDY	27.1	17.8	20.1	31.5
REVERSE PERPLEXITY	27.3	17.9	20.2	**31.8**
ALPHABET	26.7	18.2	20.3	31.2

Table 7: QA performance on AmbigQA dataset. The table is formatted the same as Table 4.

		S				
		GREEDY	REVERSE GREEDY	PERPLEXITY	REVERSE PERPLEXITY	ALPHABET
$\mathcal{D}_e$	$\text{AmbigQA}_{\text{dev}}$	60.3 / 58.3	43.7 / 42.2	68.8 / 58.1	49.5 / 41.9	75.5 / 50.5
	$\text{QAMPARI}_{\text{dev}}$	62.8 / 52.1	39.0 / 39.6	60.0 / 55.1	52.1 / 44.9	87.8 / 52.0
	$\text{QAMPARI}_{\text{test}}$	63.1 / 52.4	39.7 / 39.1	61.8 / 56.7	52.1 / 39.1	85.4 / 47.3
	$\text{QUEST}_{\text{dev}}$	70.5 / 49.1	44.0 / 42.5	60.0 / 57.1	53.1 / 42.9	91.1 / 67.6
	$\text{QUEST}_{\text{test}}$	75.3 / 57.5	46.3 / 45.5	60.1 / 54.0	50.6 / 46.0	92.3 / 51.6
Average		66.4	42.5	62.1	51.5	**86.4**

Table 8: Percentage of generated answer ordering matching in-context examples answer ordering, where we use OPT (13B) model for $\mathcal{M}$. The table is formatted the same as Table 3.

AmbigQA	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$	# pred ans
RANDOM	13.1	10.3	10.7	19.4	2.37
GREEDY	13.1	10.3	10.7	**19.5**	**2.48**
PERPLEXITY	12.9	10.0	10.5	19.2	2.44
REVERSE GREEDY	12.9	9.9	10.5	19.1	2.30
REVERSE PERPLEXITY	13.2	**10.7**	**11.0**	19.3	2.37
ALPHABET	**13.5**	10.6	**11.0**	19.3	**2.48**
QAMPARI	P_{EM}	R_{EM}	$F1_{EM}$	$F1_{F1}$	# pred ans
RANDOM	14.2 / 15.5	7.5 / 7.2	8.1 / 8.2	18.6 / 17.1	4.98 / 4.99
GREEDY	14.0 / 14.9	7.5 / 7.6	7.9 / 8.4	18.6 / **17.8**	5.66 / 5.60
PERPLEXITY	**14.7** / 15.6	7.8 / 7.7	8.3 / 8.5	19.0 / 17.6	5.46 / 5.14
REVERSE GREEDY	14.5 / 15.4	6.9* / 6.7	7.6 / 7.9	18.0* / 16.7	4.42 / 4.38
REVERSE PERPLEXITY	15.6* / **15.9**	7.6 / 7.2	8.4 / 8.3	**18.8** / 16.9	4.96 / 4.63
ALPHABET	14.4 / 15.0	**8.1*** / **7.9***	**8.5** / **8.9***	18.7 / 17.4	**6.07** / **5.72**

Table 9: QA performance on AmbigQA and QAMPARI for answer ordering strategies with OPT (13B) model. The table is formatted the same as Table 5.

		S				
		GREEDY	REVERSE GREEDY	PERPLEXITY	REVERSE PERPLEXITY	ALPHABET
$\mathcal{D}_e$	AmbigQA$_{\text{dev}}$	69.6 / 68.9	33.7 / 32.8	70.2 / 70.5	68.9 / 29.5	83.6 / 62.5
	QAMPARI$_{\text{dev}}$	63.2 / 59.8	40.7 / 40.3	57.0 / 57.3	57.3 / 42.7	92.6 / 65.9
	QAMPARI$_{\text{test}}$	61.2 / 61.4	43.5 / 43.2	57.5 / 56.4	57.5 / 43.6	92.7 / 60.7
	QUEST$_{\text{dev}}$	55.4 / 52.6	39.3 / 40.2	59.1 / 57.4	57.1 / 42.6	88.5 / 59.7
	QUEST$_{\text{test}}$	56.8 / 54.0	38.9 / 40.1	56.9 / 56.4	56.4 / 43.6	86.7 / 60.9
	Average	61.2	39.2	60.1	59.4	88.8

Table 10: Percentage of generated answer ordering matching in-context examples answer ordering, where we employ **random** in-context examples instead of most similar examples. The table is formatted the same as Table 3.

	AmbigQA$_{dev}$			*QAMPARI$_{dev}$*			*QAMPARI$_{test}$*		
	$F1_{EM}$	$F1_{F1}$	# pred ans	$F1_{EM}$	$F1_{F1}$	# pred ans	$F1_{EM}$	$F1_{F1}$	# pred ans
RANDOM	17.8	**28.7**	2.07	**9.8**	20.2	3.77	**10.0**	**19.1**	3.74
GREEDY	17.4	27.8	2.12	9.6	19.9	4.42	9.3	17.7	4.43
PERPLEXITY	**17.9**	28.3	2.11	9.7	20.0	3.99	9.7	18.6	4.03
REVERSE GREEDY	17.6	28.3	2.08	**9.8**	**20.4**	3.82	9.6	18.5	3.61
REVERSE PERPLEXITY	**17.9**	28.4	2.11	9.3	19.7	3.83	9.6	18.4	3.81
ALPHABET	**17.9**	28.5	**2.22**	**9.8**	19.8	**5.48**	9.6	17.5	**5.41**

	QUEST$_{dev}$			*QUEST$_{test}$*		
	$F1_{EM}$	$F1_{F1}$	# pred ans	$F1_{EM}$	$F1_{F1}$	# pred ans
RANDOM	4.4	12.9	3.42	3.5	11.2	3.41
GREEDY	**4.7**	12.5	4.51	3.4	10.9	4.49
PERPLEXITY	**4.7**	**13.0**	3.60	3.4	11.1	3.62
REVERSE GREEDY	4.0	12.5	3.51	3.3	11.1	3.11
REVERSE PERPLEXITY	4.6	12.6	3.09	**3.6**	**11.4**	3.28
ALPHABET	4.5	11.2	**5.84**	3.0	9.4	**5.99**

Table 11: QA performance for answer ordering strategies with random in-context examples.

Question	Who is the current chairman of african union commission?
Gold Answers	Jean Ping, Moussa Faki, Nkosazana Clarice Dlamini-Zuma
Prompt	Question: Who is the chairman of the federal reserve?\nAnswers: Alan Greenspan | Ben Bernanke | Janet Yellen\n\nQuestion: Who is the president of south africa now?\nAnswers: Thabo Mvuyelwa Mbeki | Kgalema Petrus Motlanthe | JZ\n\nQuestion: Who is the present chairperson of national human rights commission in india?\nAnswers: Justice K. G. Balakrishnan | H. L. Dattu | Cyriac Joseph\n\nQuestion: Who appoints the chairman of the finance commission?\nAnswers: the President | Pranab Mukherjee | Ram Nath Kovind | Pratibha Devisingh Patil\n\nQuestion: Who is the chairman of national commission for woman of india?\nAnswers: Lalitha Kumaramangalam | Mamta Sharma | Girija Vyas\n\nQuestion: Who is the current chairman of african union commission?\nAnswers:
Output	Jean Ping | Nkosazana Dlamini-Zuma | Moussa Faki Mahamat\n

Table 12: Prompt example of AmbigQA

Question	What movies did Scott Z. Burns screenwrite?
Gold Answers	Contagion, No Time to Die, Pu-239, Side Effects, The Bourne Ultimatum, The Informant!, The Laundromat, The Mercy, The Report
Prompt	Question: Which film has Edward Burns as a member of its cast and had Edward Burns as screenwriter?\nAnswers: Ash Wednesday | Purple Violets | She's the One | Sidewalks of New York | The Brothers McMullen | The Groomsmen | Newlyweds | Looking for Kitty | No Looking Back\n\nQuestion: Scott Ziehl was a director for what genre of film?\nAnswers: action film | crime thriller | horror film | monster film | drama\n\nQuestion: What are the publication dates of film that had Scott Z. Burns as screenwriter?\nAnswers: 2007 | 2006 | 2009 | 2013 | 2014 | 2018 | 2019 | 2020\n\nQuestion: Who are the cast members of film that had Scott Z. Burns as screenwriter?\nAnswers: Christoph Waltz | Patton Oswalt | Marsha Stephanie Blake | Jocko Sims | David Costabile | Dan Fredenburgh | Ted Levine | Ken Stott | Steven Berkoff\n\nQuestion: What are the genres of film that had Scott Z. Burns as screenwriter?\nAnswers: action film | thriller film | comedy film | spy film | drama\n\nQuestion: What movies did Scott Z. Burns screenwrite?\nAnswers:
Output	Contagion | Side Effects | The Bourne Ultimatum | The Bourne Supremacy | The Bourne Identity\n

Table 13: Prompt example of QAMPARI

Question	Mary Stewart novels
Gold Answers	A Walk in Wolf Wood, Airs Above the Ground, My Brother Michael, The Crystal Cave, The Hollow Hills, The Ivy Tree, The Last Enchantment, The Prince and the Pilgrim, The Wicked Day, The Wind Off the Small Isles, This Rough Magic, Thornyhold, Touch Not the Cat, Wildfire at Midnight
Prompt	Question: 1740s fantasy novels, 1747 books, or novels by Denis Diderot\nAnswers: Niels Klim's Underground Travels | The Sofa: A Moral Tale | Jacques the Fatalist | Rameau's Nephew | La Religieuse | The Art of Cookery Made Plain and Easy | Letters from a Peruvian Woman | Madame de La Carli\u00e8re | The Indiscreet Jewels | The Skeptic's Walk | Zadig | Mystification\n\nQuestion: Sarah Weeks novels\nAnswers: So B. It | Jumping the Scratch\n\nQuestion: 1858 British novels, 1872 fantasy novels, or Novels by George MacDonald\nAnswers: At the Back of the North Wind | Alec Forbes of Howglen | The Princess and the Goblin | David Elginbrod | The Day Boy and the Night Girl | Ranald Bannerman's Boyhood | The Princess and Curdie | Through the Looking-Glass | Phantastes | My Lady Ludlow | Sir Gibbie | Mary Marston | Doctor Thorne | Lilith | Poor Dionis\n\nQuestion: Mark Clapham novels\nAnswers: The Taking of Planet 5 | Twilight of the Gods | The Book of the War | Beige Planet Mars | Hope\n\nQuestion: Marlon James Novels\nAnswers: A Brief History of Seven Killings | John Crow's Devil | The Book of Night Women\n\nQuestion: Mary Stewart novels\nAnswers:
Output	The Crystal Cave | The Hollow Hills | The Last Enchantment | The Wicked Day\n

Table 14: Prompt example of QUEST

Question	Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?
Logical reasoning and Answer	If each chicken eats 3 cups of feed per day, then for 20 chickens they would need 3*20=«3*20=60»60 cups of feed per day.\nIf she feeds the flock 15 cups of feed in the morning, and 25 cups in the afternoon, then the final meal would require 60-15-25=«60-15-25=20»20 cups of chicken feed.\n#### 20
Prompt	Question: Mabel lives 4500 steps directly east of Lake High school. Helen lives 3/4 the number of steps that Mabel lives, directly west of the school. What's the total number of steps Mabel will walk to visit Helen so that they can do their assignments together?\nAnswer: Helen lives 3/4 * 4500 = «3/4*4500=3375»3375 steps directly west of Lake High. To reach Helen, Mabel would have to walk to 4500 + 3375 = «4500+3375=7875»7875 steps. #### 7875\n\nQuestion: Mark is 7 years older than Amy, who is 15. How old will Mark be in 5 years?\nAnswer: Mark is 15 years + 7 years = «15+7=22»22 years old. In 5 years, he will be 22 years + 5 years = «22+5=27»27 years old. #### 27\n\nQuestion: Steve has 2 boxes of pencils with 12 pencils in each box. He gave Matt 3 more pencils than he gave to Lauren. If Steve gave 6 pencils to Lauren, how many pencils does he have left?\nAnswer: Steve started with 2 * 12 = «2*12=24»24 pencils. He gave Matt 6 + 3 = «6+3=9»9 pencils. After giving away the pencils, Steve will have 24 \u2013 9 \u2013 6 = «24-9-6=9»9 pencils left. #### 9\n\nQuestion: Mandy researched 42 med schools. She applied to 1/3 of the schools she researched and got into half of the schools where she applied. How many schools was Mandy accepted to?\nAnswer: First find the number of schools Mandy applied to: 42 med schools / 3 = «42/3=14»14 med schools Then divide that number by 2 to find the number of schools where she was accepted: 14 med schools / 2 = «14/2=7»7 med schools #### 7\n\nQuestion: Rachel is stuffing envelopes. She has eight hours to complete the task, and there are 1,500 envelopes. In the first hour, Rachel stuffs 135 envelopes. The second hour she stuffs 141 envelopes. How many envelopes will Rachel need to stuff per hour to finish the job?\nAnswer: Rachel has 1500 - 135 - 141 envelopes = «1500-135-141=1224»1224 envelopes remaining to stuff. Rachel has 8 hours - 2 hours = «8-2=6»6 hours left to finish the task. Rachel needs to stuff 1224 envelopes / 6 hours = «1224/6=204»204 envelopes per hour. #### 204\n\nQuestion: Samson is going to another town which is 140 km away. He will use his car that uses ten liters of gasoline for a distance of 70 km. How many liters of gasoline will Samson need for a one-way trip?\nAnswer: Samson will need 140 km/70 km = «140/70=2»2 ten liters of gasoline for a one-way trip to a town. Therefore, he will need a total of 2 x 10 liters = «2*10=20»20 liters of gasoline. #### 20\n\nQuestion: Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?\nAnswer:
Output	Wendi gives her chickens 15 cups of feed in the morning and 25 cups of feed in the afternoon. She needs to give her chickens another 20 cups of feed in the final meal of the day. #### 20\n

Table 15: Prompt example of GSM8K