

Learning and Evaluating Factual Clarification Question Generation Without Examples

Matthew Toles¹, Yukun Huang^{1,2}, Zhou Yu^{1,3},
¹Columbia University, ²Duke University, ³Arklex.ai,
Correspondence: mt3639@columbia.edu

Abstract

Real-world tasks such as giving legal or technical advice often depend on context that is initially missing at the outset. The ability to derive missing factual information by asking clarifying questions (ACQ) is an important element of real-life collaboration on such reasoning tasks. Although intent disambiguation has been heavily investigated, factual reasoning remains underexplored. To enable evaluation of factual domain clarification question generation, we present a new task that focuses on the ability to elicit missing information in multi-hop reasoning tasks. We observe that humans outperform GPT-4o by a large margin, while Llama 3 8B Instruct does not even beat the dummy baseline in some metrics. Finally, we find that by fine-tuning Llama 3 8B Instruct on its own generations filtered via rejection sampling, we can improve information recovery by 27.6% without using any manually labeled data.

1 Introduction

In many real-world scenarios, the initial context is often incomplete, making it risky to provide answers without first seeking clarification. For instance, legal, medical, and technical advice typically depends on specific details about the individual’s situation. As language models (LMs) are increasingly used in open-domain assistant roles, their ability to clarify and gather relevant facts before offering advice is becoming more crucial.

Evaluating clarification question generation is not straightforward. Many question generation tasks evaluate generated questions based on word overlap with a ground truth label (Rahmani et al., 2023), ignoring whether the question actually acquires useful information or how difficult it is to answer. Other tasks such as those by Rao and Daumé III (2019) use human evaluators to judge the quality and informativeness of ques-

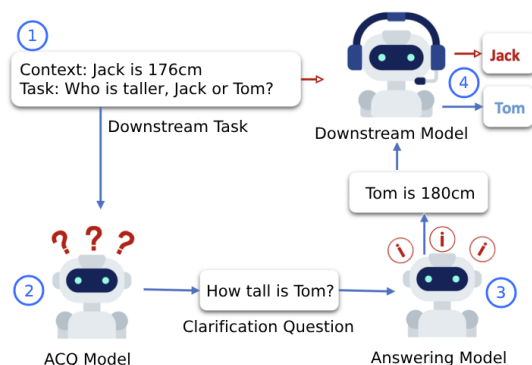


Figure 1: Overview of the HotpotQA-FLM task, which simulates the need to formulate a question. Conventionally, the downstream model performs the downstream task directly (→). However, in HotpotQA-FLM (→), critical information is missing (1). To acquire that information, the ACQ model (2) first uses the context to generate a clarification question. The question is presented to the contextually knowledgeable answering agent (3), which generates a response. The response is sent as additional context to the downstream model (4). For strong ACQ models, we expect the downstream model to achieve better performance on context + answering agent response than on context alone.

tions, but human annotation is impractical for large-scale language model benchmarking in the style of BIG-bench (Srivastava et al., 2022) and MMLU (Hendrycks et al., 2020).

Recently, some ACQ tasks including those by Zhang and Choi (2023) avoid these limitations by measuring the effect of clarifications on a downstream task. In this paradigm, which we refer to as pragmatic evaluation, an answering agent is used to dynamically generate answers to clarifying questions (Figure 1). The downstream task (e.g. QA), is then performed with and without the clarification. Pragmatic evaluation captures the objective value of the information gained while also permitting automatic evaluation.

Although underexplored in evaluations, failing to clarify basic facts in high-stakes applications

can cause serious harm to users and others. If a user asks how to clean up a chemical spill, clarifying what chemical is critically important; applying water to an alkali metal can cause explosion, but sweeping up fine powders can aerosolize toxins. Absorbing oxidizers with paper towels, however, may cause spontaneous combustion (ACS, 1995). Analogous scenarios exist in medical, legal, security, or other domains where failing to clarify can have serious real-world consequences. Although our contributions address one specific scope, we find that current models struggle to clarify key facts even in this constrained trivia QA domain. This suggests more work is necessary before models can adapt to under-specified high-stakes environments.

Compared to ambiguity in user intent, ambiguity in relevant facts poses unique challenges. Although users can generally answer questions about their own intent, they may not always know the answer to factual questions. Factual questions should be phrased to require minimal effort of recall while still learning facts relevant to the downstream task (Did you earn more than \$X? vs. Exactly how much did you earn?).

Additionally, when evaluating clarification questions in the factual domain, one must ensure critical pieces of the puzzle are not guessable or leaked in some other way. A task that nominally requires clarification ("Napoleon Bonaparte was 167cm. Who is taller, Shaquille O'Neal or Napoleon Bonaparte?") becomes trivial if the downstream agent is aware that Shaquille O'Neal was a very tall basketball player.

To bridge this gap, we introduce the PACQ task that focuses on evaluating models' ability to ask questions seeking objective factual information. Our first contribution is HotpotQA-FLM. In this task, an LLM must assist a downstream agent in answering a trivia question that is conditional on an unknown fact. The LLM must identify what information is missing, and ask for it from a third answering agent. HotpotQA-FLM prompts are created by deleting one fact from the context necessary to perform a downstream multi-hop QA task from the HotpotQA dataset (Yang et al., 2018). We term this process fact-level masking (FLM). Clarifying questions are submitted to an answering agent. The answering agent responds with one of many topically similar answers. Last, performance on the downstream task is assessed with and without the clarification.

We find state-of-the-art models struggle with

HotpotQA-FLM as compared to humans. Questions by GPT-4o recover only 48% of missing information compared to those by humans. Smaller, open source models achieve only 14% of human performance.

Given weak zero-shot performance on HotpotQA-FLM, we also contribute a method for training models to ask informative clarification questions. Notably, HotpotQA-FLM does not include examples of clarifying questions for supervised fine-tuning, which are rarely available. Instead, we train our model, Alexpaca,¹ by creating a synthetic dataset through repeated interaction with the answering agent. The dataset is filtered with rejection sampling to only include clarifying question examples that result in the expected useful response. Last, Alexpaca is fine-tuned on the synthetic dataset. Alexpaca shows a 28% increase in performance over its zero-shot Llama 3 8B Instruct source model on the full dataset. This demonstrates small models' ability to self improve at clarifying question generation given effective feedback. Alexpaca also demonstrates a scalable and cheap proof-of-concept for approaching factual ACQ tasks. The training method is suitable where supervised examples are unavailable or proprietary models perform poorly (as we find) or are unacceptable for cost, privacy, or latency reasons.

To summarize, our contributions are: 1) HotpotQA-FLM, a clarification question generation task evaluated based on objective information gain in the factual domain; and 2) Alexpaca, a rejection-sampling approach to fine-tuning models for clarification question generation not reliant on manual annotation.

2 Related Work

2.1 General Question Generation

Question Generation (QG), speaking generally, is the task of automatically generating questions (Rus et al., 2008). Questions can be generated using syntactic (Gates, 2008; Yao et al., 2012) or neural (Chen et al., 2018) approaches. Duan et al. (2017) and Wang et al. (2020) generate questions for data augmentation for QA tasks and pretraining, respectively, using convolutional, recurrent, and transformer architectures. Chatbots designed for social dialogue may ask questions to exhibit emotional intelligence, prompt users, and drive engagement

¹In honor of *Jeopardy!* host Alex Trebek (1940–2020)

(Shum et al., 2018). Question-asking can also be used for educational purposes (Kurdi et al., 2020). Four automatically evaluated question generation tasks appear in BIG-bench (Srivastava et al., 2022) including Twenty Questions, Forecasting Subquestions, Question-Answer Generation, and Question Selection.

2.2 Asking Clarifying Questions

Asking clarifying questions (ACQ) is a type of QG for acquiring additional factual knowledge or disambiguating user intent, as in (Aliannejadi et al., 2019). During general QG, outputs are often evaluated based on the Bleu, Rouge, or other word overlap metrics, as in (Qi et al., 2020; Xu et al., 2019; Min et al., 2020; Deng et al., 2022; Gaur et al., 2022; Chen et al., 2018; Meng et al., 2023) (Kostric et al., 2024) (Ang et al., 2023). Other research uses human evaluations, (Pyatkin et al., 2022; Rao and Daumé III, 2019, 2018; Chen et al., 2022). Pragmatic asking clarifying questions (PACQ), on the other hand, evaluates a question based on the usefulness of the answer it prompts (Figure 1). (Zhang and Choi, 2023; Lee et al., 2023) and (Andukuri et al., 2024) explore ACQ pragmatically but in the intent rather than factual domain. GuessWhat?! (De Vries et al., 2017), CLEVR Ask (Matsumori et al., 2021), and White et al. (2021) explore constrained iterative binary PACQ tasks in the vision domain. We present a new task specifically addressing question generation for multi-hop factual reasoning.

2.3 Related Tasks

In task-oriented dialog (TOD), the system is designed to converse with the user to perform a slot-filling task. Slot-filling tasks are typically straightforward and well-defined, like booking a hotel. Unlike in our task, the missing information, such as the desired price range, is usually clearly defined by which slots are empty (Budzianowski et al., 2018). By decoupling TOD from a fixed slot ontology and accounting for incomplete user knowledge, PACQ can be viewed as a generalization of the dialog planning and natural language generation steps of TOD. Finally, PACQ is similar to the idea of agent tool-use, where agents (Yao et al., 2023) can consult APIs like a calculator, search engine, or QA model to improve performance on a downstream task. Tool-use models like Toolformer (Schick et al., 2023) call APIs internally during generation to gather additional knowledge. Framing PACQ

as a distinct task may improve data efficiency in training and granularity of evaluation as compared to end-to-end tool use.

3 Methods

3.1 Problem Description

The goal of pragmatic asking of clarifying questions is for the ACQ model to transfer information from a knowledgeable answering agent to an executive downstream model by asking a clarifying question. In our setup the answering agent is a language model, but could also be a database, human expert, or the user. The downstream model is a model that directly executes some task for the user, such as a legal assistant chatbot or QA model. The answering agent is an agent capable of answering clarification questions related to the downstream task. This could be a human user, expert, or LLM stand-in like Flan-T5 (Chung et al., 2022). The ACQ model is a language model agent capable of generating questions that assist the downstream model in its task. It takes the downstream task as input and generates a question for the answering agent. The answering agent response is concatenated to the original context and then passed to the downstream model, giving the downstream model access to the information requested in the question. The ACQ model’s performance is evaluated using the difference between the downstream model’s performance with and without the answering agent’s answer.

Our setup, as described above and similar to (Lee et al., 2023), consists of a downstream model, D , tasked with performing some task, and an answering agent, A , which responds to questions generated by the ACQ model, C . In the next section, we present a specific $C \rightarrow A \rightarrow D$ setup and dataset on which to evaluate it.

3.2 Model Training

Creating examples of good clarification questions is expensive and challenging because question usefulness depend on the properties of the answering and downstream agents. Any change to these agents may require a different question generation strategy. Therefore, it is useful for models to be trained through interaction with the answering agent rather than through manual supervision. We propose a method where a zero-shot model repeatedly generates clarifying questions, and is then fine-tuned on only the clarifying questions that pro-

duce useful information.

3.3 Problem Definition

Let t be a natural language statement of a task. Let the context for the task be comprised of f_1, \dots, f_n natural language facts. Let example $x = t + f_1 + \dots + f_n$, where $+$ indicates string concatenation. Let $D(x) \rightarrow y$ be a downstream model that takes x as input and outputs y . Let $C(x) \rightarrow q$ be an ACQ model that takes x as input and generates a natural language question q . Let $R(D, x, y) \rightarrow r$ be some reward on which D is evaluated, where more positive values are better, such as F-score, accuracy, or negative loss. For brevity, we often omit D and y .

We say a fact f is supporting if it is believed that $R(x + f) > R(x - f)$, where $-$ represents deletion (if present). Otherwise we say f is distracting (Yang et al., 2018). Let $A(q) \rightarrow f_r$ be an answering agent that takes q as input and returns a response f_r . The PACQ task is to create a model C that maximizes

$$\Delta r = R(x + f_r) - R(x)$$

One may construct more complex versions of PACQ involving multiple missing facts, iterative asking, multiple answering agents, or cost functions for different types of questions. In this paper, we limit PACQ to the costless, single-mask, single-turn, single-answering agent case and we do not address determining whether a task lacks sufficient context.

4 Experiments

4.1 Dataset

We contribute HotpotQA-FLM, a version of the QA dataset HotpotQA for evaluating pragmatic asking of clarifying questions (Yang et al., 2018). HotpotQA is a multi-hop QA reasoning task where each example contains both supporting and distractor facts from Wikipedia as determined by human annotators. We choose reward function R to be the F1 score of the word overlap between the predicted answer and the ground truth answer following the original HotpotQA. Thus $r \in [0, 1]$ and $\Delta r \in [-1, 1]$.

To evaluate our ACQ model, we create three context examples: the incomplete example x^i missing some context, the complete example x^c with full context, and x^r which contains the incomplete context plus additional context derived from the clari-

fying question. The incomplete and complete contexts will serve as the worst- and best-case benchmarks against which we compare the response context.

First, we obtain x^c which contains the task and every supporting fact (Figure 2) from HotpotQA. Next, we apply fact-level masking to each HotpotQA example, where facts are helpfully provided as a list. From each complete example, we create an incomplete example x^i by randomly selecting one supporting fact, f^* , to be the masked fact and deleting it from the context: $x^i = x^c - f^*$. When missing one supporting fact, the downstream task becomes substantially more difficult, even for strong zero-shot models like GPT-4o (OpenAI, 2024) (Figure 5). The masked fact, along with the distractor facts and the other supporting facts, make up the set of responses, f_r , the answering agent may give. Finally, we prompt the question model with the incomplete context to generate a question, then generate a response f_r from the answering agent. We create the response example x^r by appending $x^r = x^i + f_r$. To benchmark human performance, one author of this paper annotated a test set of 400 clarifying questions from examples also included in the full set.

In general, we expect the complete example x^c , which contains every supporting fact, to have the highest possible reward. Meanwhile, we say an example x is improvable if there exists at least one possible response f_r such that $\Delta r(f_r) > 0$. By masking facts in x^c we can decrease the reward on the example, producing an improvable self-supervised example. Note that not all incomplete examples will be improvable, such as when:

- Two facts contain redundant information
- D has memorized knowledge of information in f^*
- f^* is mislabeled as supporting
- x^i still allows D to make a spurious correlation without f^*

It is also possible for x^i to be improved by a response f_r even if $f_r \neq f^*$, if f_r and f^* contain similar information. We automatically compute Δr on the full and test sets using fact-level masking, finding that 27.6 and 28.5% of examples, respectively, are improvable. We preserve unimprovable examples in the dataset to avoid bias; the downstream model may sometimes achieve the correct response through a spurious correlation on the incomplete example, but fail to make the spurious correlation after receiving the response. Similarly, the down-

t	When was the composer of "Persian Surgery Dervishes" born?	
f_1^{sup}	Persian Surgery Dervishes is a recording of two live solo electric organ concerts, the first held in Los Angeles on 18 April 1971 and the second in Paris on 24 May 1972, by avant-garde minimalist composer Terry Riley.	Incomplete Example x^i
$f_2^{sup} (f^*)$	Terrence Mitchell "Terry" Riley (born June 24, 1935) is an American composer and performing musician associated with the minimalist school of Western classical music.	Complete Example x^c
f_1^{dis}	Thomas Christian David (December 22, 1925 - January 19, 2006) was an Austrian composer, conductor, choral conductor, and flutist.	Candidate Oracle Responses
f_2^{dis}	Abdolreza Razmjoo is a composer, arranger and singer Tenor of Iran Kurdish ancestry from Kermansha.	

Figure 2: An example containing a downstream task t , supporting facts $f_{1,\dots,n}^{sup}$, and distractor facts $f_{1,\dots,n}^{dis}$. (Additional facts not shown.) We create an **incomplete example** x^i by masking one supporting fact, f^* , chosen at random, from the facts in the **complete example** x^c . Prompted with x^i , the ACQ model poses a question to the answering agent which returns one **answering agent response** f_r from the supporting or distractor facts. We then append $x^r = x^i + f_r$, which we expect to improve downstream model performance $D(\cdot)$

stream model may fail even given the masked fact, but succeed given another fact if the other fact contains more helpful information.

4.2 Evaluation Implementation Details

To generate and evaluate answers to PACQ questions, we construct the following pipeline. The ACQ model C takes an incomplete example x^i as input to generate a clarifying question q . As baselines for C we choose GPT-4o (OpenAI, 2024) and Llama 3 8B Instruct (AI@Meta, 2024). We select these models for their strong performance on zero-shot tasks. We choose a prompt template for each model by evaluating three zero-shot and three 5-shot in-context prompts on 400 examples from the training dataset 8.1. In addition, we create a new model, Alexpaca, by fine tuning Llama 3 on a dataset of its own generations filtered with rejection sampling. Finally, we include a dummy Repeater model among the baselines, which simply returns the input task.

Questions generated by C are passed to the answering agent A , a Flan-T5-Base model, which we choose for its accessibility and strong zero-shot performance on other QA tasks (8.2). The answering agent serves as a stand-in for a human expert answering clarifying questions generated by C . A returns f_r , the most likely response to q from among all possible distractor facts F^{dis} present in the original HotpotQA example, all supporting F^{sup} facts, $n - 1$ of which are already present in the context, and the masked fact f^* . HotpotQA examples contain, on average, 39.2 distractor facts (standard deviation 11.4) and 2.43 supporting facts

(standard deviation 0.71).

To create the response example x^r , we append the answering agent response to the incomplete example. Note that by appending rather than inserting, the order of facts may be altered as compared to x^c , even if $f_r = f^*$, which may occasionally affect the output of the downstream model.

Finally, we compare the performance of the downstream model, D , given contexts x^i , x^r , and x^c . D is also a Flan-T5-Base model (8.3). We choose Flan-T5-Base over models using more parameters or training data because we expect they are more likely to answer based off of context rather than information memorized from training data (e.g., Wikipedia). If C produces a question with positive utility towards D , then one should expect $R(x^c) \geq R(x^r) > R(x^i)$. To express reward relative to its theoretical minimum ($R(x^i)$) and maximum ($R(x^c)$) values, we define recovery as:

$$\rho = 100 \cdot \frac{R(x^r) - R(x^i)}{R(x^c) - R(x^i)}$$

and select F1 recovery as our downstream evaluation metric.

4.3 Alexpaca: Fine-Tune through Interaction

Annotating high quality clarifying questions is challenging and costly. For this reason, we train our model, Alexpaca, purely through interacting with the answering agent. First, we use the Llama 3 8B Instruct foundational model to generate a set of clarifying question examples using rejection sampling. To ensure examples are of high quality, we

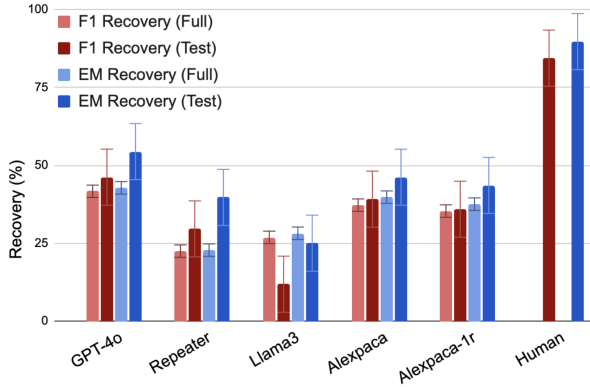


Figure 3: F1 and exact match recovery for PACQ models and human annotators. Results shown for the full validation set ($n = 7404$) and the test set ($n = 400$), which contains human-generated ACQ questions. Alexpaca-1r indicates single round rejection sampling.

reject questions if the answering agent response does not match the masked fact. We repeat the generation for each example until one is accepted, or until $k = 40$ rounds. Each round we increase generation temperature by $2/k$, starting at 0.01 in order to encourage exploration in later rounds. Finally, we fine-tune the same Llama 3 foundational model on the rejection sampling dataset.

5 Results and Discussion

5.1 Baseline Performance

We report F1 and exact match recovery results for ACQ models on the full HotpotQA validation set ($n = 7404$, Figure 3). Of all models, GPT-4o performs best in both F1 and exact match (EM), recovering 41.7% and 42.8% respectively. These results, however, fall short of complete recovery of missing information, indicating room for improvement even in strong zero-shot models. Other models perform substantially worse. Llama 3 achieves 26.9% F1 recovery, which is only a moderate improvement over the dummy Repeater model. We suspect Repeater achieves its positive recovery (22.5%) by exploiting a bias in the answering agent towards choosing responses with high keyword overlap with the input question.

5.2 Alexpaca Fine-Tuning Performance

Alexpaca exceeds baseline Llama 3 performance by 37.2% vs. 26.9 F1 recovery ($p = 0.00074$), demonstrating a method for self-improving ACQ models given an answering agent rather than example clarifying questions. Although GPT-4o achieves higher performance than Alexpaca, Alexpaca is

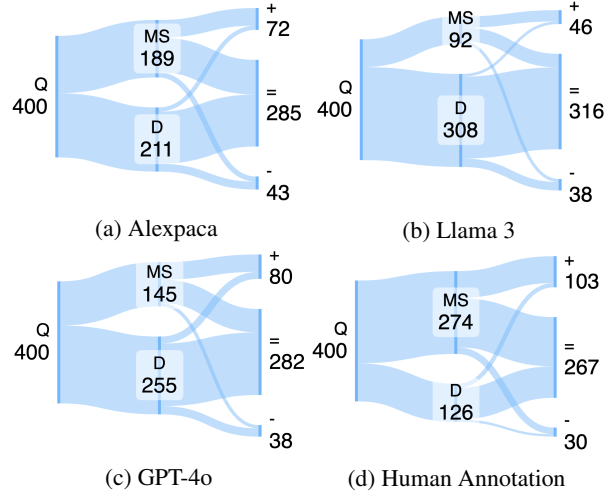


Figure 4: Proportion of questions (Q) answered with a masked fact (MS) vs. distractor (D) by answering agent (middle values). Proportion of answers given resulting in positive, zero, or negative difference in downstream model performance (right values).

open-source and uses many times fewer parameters compared to GPT-4o. Alexpaca therefore may be more suitable in circumstances where cost, latency, or privacy are a concern. We report the average of results for five random seeds. During training dataset creation, repeatedly attempting to generate passing examples up to 40 times each (Alexpaca) improves F1 recovery by 6.0% points compared to using a single attempt (Alexpaca-1r). We believe that challenging examples accepted in later rounds of rejection sampling and generated at higher temperature have a disproportionate effect on model behavior.

5.3 Alexpaca Behavior

Although Alexpaca elicits the masked fact more often than GPT-4o on the test set (189 vs. 145), Alexpaca’s overall improvement rate is still lower (72 vs. 80). Likely this is an artifact of the Alexpaca training rejection criteria wherein acceptance is determined by eliciting the masked fact rather than actual downstream improvement. This indicates room for improvement in baseline models performing PACQ. Attempts to correct this bias by accepting examples based on recovery rather than masked fact response did not achieve statistically significant improvement in F1 recovery, possibly due to a lower signal-to-noise ratio in end-to-end systems.

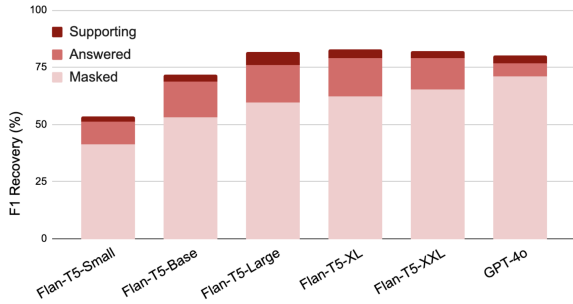


Figure 5: Supporting, answered, and masked F1 as a function of downstream model architecture.

5.4 Comparison to Human Performance

We find that human-generated questions on the test set are more likely to elicit the masked fact f^* in the response (Figure 4). Eliciting the masked sentence usually, but not always, produces as good or better a result in the downstream model compared to eliciting a distractor. This leads to human annotations performing significantly better than the best baseline models. Human annotation achieved 84.4% F1 and 89.7% EM recovery, compared to the strongest baseline, GPT-4o, which achieved 46.2% F1 and 54.4% EM recovery on the test set (Figure 3).

5.5 Downstream Model Ablation

We evaluate all available sizes of Flan-T5 and GPT-4o as candidate downstream models using a Flan-T5-Base model as the answering agent and human-generated questions as the ACQ model. Models lose between 9.2% (GPT-4o) and 22.0% (Flan-T5-Large) absolute points F1 score as a result of masking a single supporting fact (Figure 5). We suspect GPT-4o is more robust than Flan-T5 since in exploration they appear to have memorized large portions of Wikipedia, which minimizes the impact of removing Wikipedia facts from context. This makes them less well suited as indicators in the role of the downstream model compared to Flan-T5. Models recover between 62.0% (GPT-4o) and 84.4% (Flan-T5-Base) of the F1 score lost during masking after including the answering agent response to human generated questions. Although models are affected differently by FLM, with GPT-4o being the most robust, reasonable consistency in F1 recovery rate suggests that valid results could be achieved across many model choices.

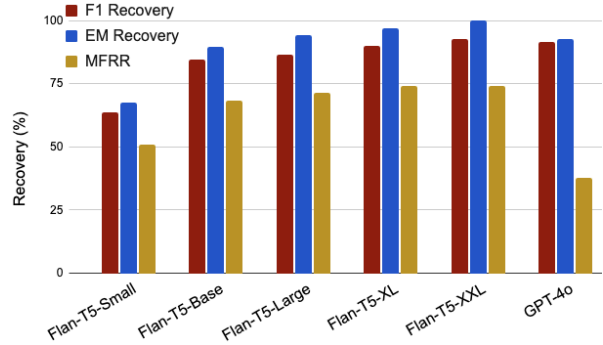


Figure 6: F1, exact match and masked fact response rate (MFRR) as a function of answering agent size and architecture.

5.6 Answering Agent Ablation

We test GPT-4o and all sizes of Flan-T5 as the Answering Agent on human-generated questions. Flan-T5-Base and larger respond with the masked fact in more than 68% of cases (Figure 6). Furthermore, we observe consistently strong performance by these models on F1 and exact match, with both metrics exceeding 84% recovery in all cases. This indicates that when prompted by well-formed and informative questions, Flan-T5 of size Base and larger can consistently respond with appropriate answers. For the sake of accessibility, we choose the smallest strong model, Flan-T5-Base, as our answering agent. Interestingly, although GPT-4o responds with the masked fact far less frequently than any Flan-T5 model (GPT-4o: 37.8%, Flan-T5-XXL: 74.0%), GPT-4o achieves the second-highest F1 recovery overall and 92.6% exact match recovery. This suggests that although GPT-4o gives distractor or redundant supporting facts most of the time, the facts it chooses still carry critical information, illustrating the importance of measuring information gain rather than nominal correctness.

5.7 Error Analysis

We observe one failure mode associated with the answering agent and three associated with the ACQ model, which prevent PACQ questions from recovering missing information. Firstly, the answering agent may return an irrelevant and unhelpful response. In 31.5% of cases, human-generated questions induce responses other than the masked fact. When $f^* \neq f_r$, the F1 score of the downstream model increases in only 11.1% of cases, compared to 32.5% of cases when $f^* = f_r$ (Figure 4d). When a distractor fact does cause an increase in F1, it is often because information in the distractor fact con-

Model	Full					Test				
	F1	F1 Recovery	EM	EM Recovery	MFRR	F1	F1 Recovery	EM	EM Recovery	MFRR
GPT-4o	61.6	41.7	45.9	42.8	24.7	61.6	46.2	48.2	54.4	36.2
Repeater	58.3	22.5	43.1	22.8	29.1	58.5	29.6	45.8	39.7	32.8
Llama 3	59.1	26.9	43.9	28.2	22.8	55.2	11.8	43.3	25.0	23.0
Alexpaca	60.9	37.2	45.5	39.8	39.2	60.3	39.1	46.9	46.2	45.3
Alexpaca-1r	60.5	35.3	45.2	37.5	37.9	59.7	35.9	46.4	43.5	43.1
Human	-	-	-	-	-	68.8	84.4	54.3	89.7	68.5

Table 1: F1 and exact match recovery for full validation set and manually annotated test set.

tains overlaps with information in the masked fact.

Other times, the failure mode is due to the ACQ model generating poor questions. In some examples, GPT-4o asks for information already present in the context:

Context: Roger O. Egeberg’s other roles included Assistant Secretary for Health and Scientific Affairs during the Nixon administration and Dean of the University of Southern California’s medical school.
Task: Roger O. Egeberg was Assistant Secretary for Health and Scientific Affairs during the administration of a president that served during what years?
GPT-4o: Under which president did Roger O. Egeberg serve as Assistant Secretary for Health and Scientific Affairs? ✗
Human: During what years did Nixon serve as president? ✓

In other cases, GPT-4o appears to struggle with simple categorical reasoning:

Context: Backford Cross is located around the A41/A5117 junction, south of Great Sutton and about 1.5 mi north of the village of Backford, near Chester.
Task: What major truck road is located in Backford Cross?
GPT-4o: What are the major roads near Backford Cross? ✗
Human: Is the A41 or the A5117 a major truck road? ✓

Alexpaca makes mistakes similar to GPT-4o, as well as hallucinating relevant details to ask about:

Context: Dinosaur is a 2000 American CGI animated adventure film produced by Walt Disney Feature Animation and The Secret Lab and released by Walt Disney Pictures.
Task: Which came out first, Dinosaur or McFarland, USA
Alexpaca: What Disney animated film was released in 1942? ✗
GPT-4o: When was McFarland, USA released? ✓

Since the Alexpaca training approach does not focus on style improvement, Alexpaca sometimes retains Llama 3’s prototypical cheery chatbot verbosity, though more frequently asks informative questions:

Context: Sir Arthur Ignatius Conan Doyle (22 May 1859 – 7 July 1930) was a British writer best known for his detective fiction featuring the character Sherlock Holmes.
Task: Who was born first, Arthur Conan Doyle or Penelope Lively?
Llama 3: A clever question! To answer the final question, I can ask: "When was Arthur Conan Doyle born?" ✗
Alexpaca: A clever question!
 To answer this question, I can ask: "When was Penelope Lively born?"
 This question will help me determine the answer to the original question by allowing me to compare the birth dates of Arthur Conan Doyle (22 May 1859) and Penelope Lively (1947). ✓

Although verbose, Alexpaca’s question does in fact produce the masked sentence, also illustrating the answering agent’s robustness to noise and style. Ironically, Alexpaca identifies that Penelope Lively’s birthdate (17 March 1933) is missing, then hallucinates it to be in 1947.

6 Conclusion

In this paper, we presented a benchmark for objectively evaluating clarifying questions and observed that state-of-the-art zero-shot LLMs struggle at this task compared to humans. To overcome these challenges, we introduced fact-level masking and HotpotQA-FLM, a self-supervised PACQ dataset, and an associated evaluation process. Finally, we demonstrated a training method for the Alexpaca model that relies on agent-agent interaction rather than supervised examples of clarifying questions to self-improve over baseline.

7 Limitations

One limitation of the Alexpaca approach is that it requires answering agent responses to be labeled as useful or not useful. The FLM process produces such labels implicitly. In the real-world, however, whether classifying answers is more practical than annotating clarification questions examples depends on the situation. We also note the limited scope of our benchmark, which addresses only two- or three-hop trivia-style questions. Similarly,

subjective situations and those contingent on user intent are not included. Nonetheless, we believe this dataset and approach lead to improve factual clarification question generation in language models and LLM safety in high-stakes, ambiguous environments.

Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. (DGE-2036197).

References

- ACS. 1995. Guide for chemical spill response. Available at <https://www.acs.org/about/governance/committees/chemical-safety/publications-resources/guide-for-chemical-spill-response.html> (2024/08/13).
- AI@Meta. 2024. [Llama 3 model card](#).
- Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 475–484.
- Chinmaya Andukuri, Jan-Philipp Fränken, Tobias Gerstenberg, and Noah D Goodman. 2024. Star-gate: Teaching language models to ask clarifying questions. *arXiv preprint arXiv:2403.19154*.
- Beng Heng Ang, Sujatha Das Gollapalli, and See Kiong Ng. 2023. Socratic question generation: A novel dataset, models, and evaluation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 147–165.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. Multiwoz—a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*.
- Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. 2018. LearningQ: a large-scale dataset for educational question generation. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12.
- Jifan Chen, Aniruddh Sriram, Eunsol Choi, and Greg Durrett. 2022. Generating literal and implied sub-questions to fact-check complex claims. *arXiv preprint arXiv:2205.06938*.
- Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.
- Harm De Vries, Florian Strub, Sarath Chandar, Olivier Pietquin, Hugo Larochelle, and Aaron Courville. 2017. Guesswhat?! Visual object discovery through multi-modal dialogue. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5503–5512.
- Yang Deng, Wenqiang Lei, Wenxuan Zhang, Wai Lam, and Tat-Seng Chua. 2022. Pacific: towards proactive conversational question answering over tabular and textual data in finance. *arXiv preprint arXiv:2210.08817*.
- Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.
- Donna Gates. 2008. Generating look-back strategy questions from expository texts. In *The Workshop on the Question Generation Shared Task and Evaluation Challenge, NSF, Arlington, VA*. <http://www.cs.memphis.edu/~vrus/questiongeneration//1-Gates-QG08.pdf>.
- Manas Gaur, Kalpa Gunaratna, Vijay Srinivasan, and Hongxia Jin. 2022. Iseqq: Information seeking question generation using dynamic meta-information retrieval and knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10672–10680.
- Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.
- Ivica Kostrić, Krisztian Balog, and Filip Radlinski. 2024. Generating usage-related questions for preference elicitation in conversational recommender systems. *ACM Transactions on Recommender Systems*, 2(2):1–24.
- Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30:121–204.
- Dongryeol Lee, Segwang Kim, Minwoo Lee, Hwanhee Lee, Joonsuk Park, Sang-Woo Lee, and Kyomin Jung. 2023. Asking clarification questions to handle ambiguity in open-domain qa. *arXiv preprint arXiv:2305.13808*.
- Shoya Matsumori, Kosuke Shingyouchi, Yuki Abe, Yosuke Fukuchi, Komei Sugiura, and Michita Imai. 2021. Unified questioner transformer for descriptive question generation in goal-oriented visual dialogue. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1898–1907.

- Yan Meng, Liangming Pan, Yixin Cao, and Min-Yen Kan. 2023. Followupqg: Towards information-seeking follow-up question generation. *arXiv preprint arXiv:2309.05007*.
- Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*.
- OpenAI. 2024. [Gpt-4o system card](#). *Preprint*, arXiv:2410.21276.
- Valentina Pyatkin, Jena D Hwang, Vivek Srikumar, Ximing Lu, Liwei Jiang, Yejin Choi, and Chandra Bhagavatula. 2022. Clarifydelphi: Reinforced clarification questions with defeasibility rewards for social and moral situations. *arXiv preprint arXiv:2212.10409*.
- Peng Qi, Yuhao Zhang, and Christopher D Manning. 2020. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. *arXiv preprint arXiv:2004.14530*.
- Hossein A Rahmani, Xi Wang, Yue Feng, Qiang Zhang, Emine Yilmaz, and Aldo Lipani. 2023. A survey on asking clarification questions datasets in conversational systems. *arXiv preprint arXiv:2305.15933*.
- Sudha Rao and Hal Daumé III. 2018. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. *arXiv preprint arXiv:1805.04655*.
- Sudha Rao and Hal Daumé III. 2019. Answer-based adversarial training for generating clarification questions. *arXiv preprint arXiv:1904.02281*.
- Vasile Rus, Zhiqiang Cai, and Art Graesser. 2008. Question generation: Example of a multi-year evaluation campaign. *Proceedings in the Workshop on the Question Generation Shared Task and Evaluation Challenge*.
- Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761*.
- Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From Eliza to XiaoIce: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19:10–26.
- Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, and 1 others. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *arXiv preprint arXiv:2206.04615*.
- Yanmeng Wang, Wenge Rong, Jianfei Zhang, Shijie Zhou, and Zhang Xiong. 2020. Multi-turn dialogue-oriented pretrained question generation model. *Complex & Intelligent Systems*, 6:493–505.
- Julia White, Gabriel Poesia, Robert Hawkins, Dorsa Sadigh, and Noah Goodman. 2021. [Open-domain clarification question generation without question examples](#). *Preprint*, arXiv:2110.09779.
- Jingjing Xu, Yuechen Wang, Duyu Tang, Nan Duan, Pengcheng Yang, Qi Zeng, Ming Zhou, and Xu Sun. 2019. Asking clarification questions in knowledge-based question answering. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 1618–1629.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). *Preprint*, arXiv:2210.03629.
- Xuchen Yao, Gosse Bouma, and Yi Zhang. 2012. Semantics-based question generation and implementation. *Dialogue & Discourse*, 3(2):11–42.
- Michael JQ Zhang and Eunsol Choi. 2023. Clarify when necessary: Resolving ambiguity through interaction with lms. *arXiv preprint arXiv:2311.09469*.

8 Appendix

8.1 List of Prompts

1. Ask another question that would help you answer the following question: {context} {q1}
2. Some information is missing from this context. Ask a simpler question that would help you answer it. Context: {context} Main Question: {q1} Simpler question:
3. What question can you ask to help you answer the final question? {context} {q1} You can ask:
4. Ask another question that would help you answer the following question: {in-context examples} {context} {q1}
5. Some information is missing from this context. Ask a simpler question that would help you answer it. {in-context examples} Context: {context} Main Question: {q1} Simpler question:
6. What question can you ask to help you answer the final question? {in-context examples} {context} {q1} You can ask:

Based on performance on $n = 400$ examples from the HotpotQA train dataset we select prompt 3 for Llama 3, and GPT-4o, though improvement over other prompts was not statistically significant.

8.2 Answering Agent Implementation Details

For Flan-T5 answering agents, we prompt the model with

Question: {clarifying question}\n context: {candidate answer}\n prompt: Does the context answer the question, yes or no?

We then return the answer with the highest ratio of the "yes" to "no" logits. For the GPT-4o answering agent, we prompt the model with

Question: {clarifying question}\n\n {enumerated answers} \n\n Which answer is correct? Only say the number of the answer, nothing else.

and return the answer at the index returned. If no valid index is returned, we return a random answer.

8.3 Downstream Agent Implementation Details

For downstream agents, we prompt the model with

{task} {article title 1}: {fact 1} ... {article title n}: {fact n} Answer in as few words as possible:

8.4 Answering Agent Architecture Ablation

	F1	F1 Recovery	EM	EM Recovery	MFRR
Flan-T5-Small	64.9	63.8	50.5	67.6	50.8
Flan-T5-Base	68.8	84.4	54.2	89.4	68.5
Flan-T5-Large	69.2	86.5	55.0	94.1	71.3
Flan-T5-XL	69.8	90.1	55.5	97.1	74.3
Flan-T5-XXL	70.4	92.9	56.0	100.0	74.0
GPT-4o	69.5	88.4	54.3	89.7	43.5
Incomplete	53.0	0.0	39.0	0.0	-
Complete	71.7	100.0	56.0	100.0	-

Table 2: Answering agent architecture ablation for answering agents using Flan-T5-Base as downstream model on the full validation set.

8.5 Downstream Agent Architecture Ablation

	F1				EM			
	Incomplete	Response	Complete	Recovery	Incomplete	Response	Complete	Recovery
Flan-T5-Small	41.4	51.1	53.6	79.3	28.5	35.3	37.8	73.0
Flan-T5-Base	53.0	68.8	71.7	84.4	39.0	54.3	56.0	89.7
Flan-T5-Large	59.8	76.1	81.8	74.2	42.5	58.0	63.5	73.8
Flan-T5-XL	62.3	78.9	82.9	80.5	45.8	60.8	64.8	78.9
Flan-T5-XXL	65.2	78.9	82.2	80.6	50.5	62.5	65.8	78.7
GPT-4o	70.9	76.6	80.1	62.0	34.5	38.0	39.5	70.0

Table 3: Downstream agent architecture ablation using Flan-T5 base as answering agent on the Full validation set.

8.6 Alexpaca Training Hyperparameters

Examples	500
Per Device Batch Size	2
Gradient Accumulation Steps	16
Learning Rate	2e-5
Weight Decay	0
Warmup Ratio	0.03
Learning Rate Schedule	Cosine
Data Parallel	Full Shard Auto Wrap
Random Seed	0

We perform training on 2x NVIDIA A100 GPUs. We perform inference on 1x NVIDIA RTX A6000 with batch size 1.