RESOLVING AMBIGUITY THROUGH PERSONALIZATION IN MULTI-TURN LLM CONVERSATIONS

Sophia Sun *
University of California San Diego
shs066@ucsd.edu

Abishek Sankararaman & Balakrishnan (Murali) Narayanaswamy
Amazon Web Services
Santa Clara, CA

ABSTRACT

This paper explores LLMs' ability to perform consistent personalized generation incorporating user feedback. We first show that it is challenging for LLMs to (1) utilize feedback consistently in long conversations, (2) reason about multiple partial or conflicting feedback, and (3) adapt to changing preferences within a conversation. These challenges show that input information selection is crucial for improving multi-turn LLM performance. We propose a novel solution of building a **CORESET** of past conversations, a principled approach of personalization. In addition to addressing the long history, conflict, and preference change challenges, coresets are an effective way to reduce input tokens, making these services more cost-effective. We show that our coreset algorithm improves upon state-of-the-art methods on both synthetic and real-world ambiguity datasets compared to memory and personalization benchmarks.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable abilities in addressing a wide range of tasks via natural language, which has led to the vision of their use as interactive chat assistants in many domains (Köpf et al., 2024; Ross et al., 2023). Many of these agents function in areas that utilize retrieval-augmented generation (RAG). However, LLMs face challenges in tasks where personal knowledge acquired through extended user interactions plays a significant role in completion, such as counseling or secretarial roles (Zhong et al., 2024).

We identify that many of the personalization can be in the form of disambiguation. Ambiguity is a natural phenomenon of language, allowing speakers to balance efficiency and clarity in communication (Piantadosi et al., 2012). However, ambiguity in a user's instructions can often be a challenge for LLMs when the task requires semantic parsing or reasoning. How can we provide feedback to an LLM to resolve ambiguities and make the model remember my preferences, or adapt as needed? In this work, we develop a system to address these questions.

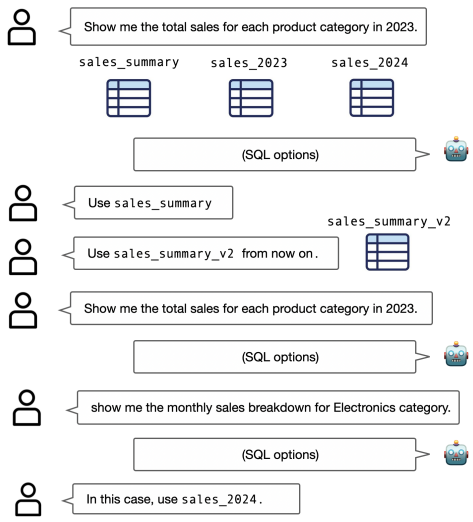


Figure 1: A real-world inspired example showing personalized disambiguation is needed for conversations involving semantically similar entities.

*Work done when interning with Amazon Web Services.

We identify that a specific kind of ambiguity often arises in QA and text2SQL. Cole et al. (2023) formulates as "denotation uncertainty", which is the ambiguity of the denotation of entities in the question. For example, when we ask "Who won the Olympics in fencing?" the "Olympics" term can mean different years, genders, and blades, resulting in a wide distribution of possibly correct answers. After we clarify the question to "Who won the gold medal in the 2020 Olympics in the men's foil?", we have eliminated the denotation uncertainty.

Note that this is different from the scenario when the question is unambiguous but the answer is uncertain, such as "what are the causes of cancer?" or "name the most livable city in 10 years." We call this type epistemic uncertainty, which can only be reduced by obtaining clearer data.

Formally, we can represent epistemic uncertainty and denotation uncertainty by factoring them as in the following equation, where q represents the user's request, a the answer, z the denoted entity (elements of the set $\mathbf{z}$, the possible things that the question can be referring to) , and $\mathcal{D}$ a set of documents provided to the model.

$$P(a|q, \mathcal{D}) = \Sigma_{z \in \mathbf{z}} P(a|z, q, \mathcal{D}) P(z|q, \mathcal{D})$$

In summary, our contributions are:

- **Formulating the Disambiguation through Personalization (DtP) problem.** (Section 3) We outline a common use case where a user will use a structured generation service such as RAG or text-to-SQL multiple times and would provide feedback.
- **Illustrate current inadequacies.** (Section 4) We conduct extensive studies on a controlled synthetic dataset, and identify current inadequacies with challenges current models face. In summary, they are (1) utilize feedback consistently in long conversations, (2) reason about multiple partial or conflicting feedback, and (3) adapt to changing preferences within a conversation.
- **CORESET algorithm to address challenges.** (Section 5) We provide detailed algorithms for our approach to address the 5 challenges, and verify our method on our synthetic dataset and a real world dataset.

2 RELATED WORK

Method ↓ Capabilities →	Ambiguity	Feedback Incorporation	Feedback Persistence	Changing Preferences
Uncertainty Quant.	✅	❌	❌	❌
Clarifying Questions	✅	✅	❌	❌
Memory	❌	✅	✅	?
Conflict Resolution	?	❌	❌	✅
CORESET (Ours)	✅	✅	✅	✅

Table 1: Comparing our method CORESET with Literature. The rows are existing methodologies in our problem space elaborated in the Related Work section, and the columns are aspects in our problem space elaborated in Section 4.

Uncertainty Quantification. Uncertainty quantification (UQ) is a crucial step for reliable and trustworthy LLM deployment (see Geng et al. (2024) for a recent survey). Recent studies show that the logits of out-of-the-box LMs tend to exhibit overconfidence, even when wrong (Desai & Durrett, 2020; Vasconcelos et al., 2023). In tasks such as code generation and knowledge extraction, detecting uncertainty can improve generation quality and mitigate potential hallucination. Commonly approaches of UQ includes self-consistency (Manakul et al., 2023; Zhang et al., 2023), prompt modification and ensembling (Tonolini et al., 2024), and leveraging surrogate models for classifiers and filters (Quach et al., 2023; Shrivastava et al., 2023).

Recent works have sought to distinguish between the two types of uncertainty: aleatoric and epistemic. Epistemic uncertainty arises from the lack of knowledge and can be decreased with knowledge, whereas aleatoric uncertainty is the inherent noise of the data and is irreducible. Hou et al. (2023) decomposes the two through adding contextual information into prompting. Ahdritz et al. (2024) shows that aleatoric uncertainty can be calibrated via linear probes. We advocate that another for looking at such distinction is *ambiguity*.

Ambiguity. While *uncertainty* is quantifying the confidence in the outcome of a task, *ambiguity* refers to the multiple possible interpretations or under-specification of the input instruction. From a linguistics perspective, ambiguity is complex - Liu et al. (2023) identified 11 different sources in natural language. For specific applications, on the other hand, researchers can prescribe the domain of ambiguity: In open-domain question answering (QA), for example, a common source is the time and location context (Min et al., 2020). In text-to-SQL (Bhaskar et al., 2023), ambiguity mostly arises from which specific column, table, or join paths are being referred. In machine translation, it can be pronoun and "it" resolution (Pilault et al., 2023).

Addressing Ambiguity. Ambiguity can be addressed by LLMs in various ways. For the goal of complete representation, Lee et al. (2024); Amplayo et al. (2022) provides long-form answers to disambiguate between entities; Sun et al. (2023); Kim et al. (2023) presents a list of answer candidates with disambiguation; Cole et al. (2023) selects answers by evaluates scores for each candidate.

To resolve ambiguity, Asking clarifying questions to allow users to specify the intended interpretation of the question (Zhang & Choi, 2023; Lee et al., 2023); we can also resolve ambiguity using tools Gou et al. (2023), by de-biasing entity popularity Chen et al. (2021), or by reasoning about environment constraints (Park et al., 2023a). Due to challenges in data collection, there are less work on disambiguation that requires multi-turn interaction with users. In previous multi-turn benchmarks Wang et al. (2023), we assume the agent giving feedback knows the ground truth answer.

Personalization. Extracting and learning user preferences have been explored in the space of LLM Agents. Chain-of-Thought (Wei et al., 2022) and Scratchpads (Nye et al., 2021) encourage the LLM to generate intermediate reasoning, using the LLM's own context as a form of working memory. Since our setting focuses on disambiguation and less on explicit memory, we refer readers to (Li et al., 2024) for a survey in the Agent space. For learning user preference, Gao et al. (2024) query the LLM to generate a preference summary that best explains user edits. Si et al. (2022) shows that LLM can learn incorporated unseen knowledge via in context learning.

Positional Bias Retrieval Augmented Generation (RAG) is an effective solution to hallucinations, and have achieved remarkable improvements by incorporating supporting knowledge into the input of LLMs. One of the main challenges of RAG is long context: in Multi-document question answering (Multi-doc QA), LLMs often fail to produce correct answers if related documents are located in the middle of the context, called "lost in the middle" (Liu et al., 2024). Approaches to solve this problem includes fine-tuning (Zhang et al., 2024) and multi-step prompting that identifies relevant documents first (He et al., 2024).

In-context Conflict Changing user preference can also be seen in the lens of in-context information conflict (see survey by Xu et al. (2024)). Previous research empirically demonstrates that the performance of a language model can be significantly influenced by the presence of misinformation Zhang & Choi (2021) or outdated information Du et al. (2022) within a specific context. Fact checking Du et al. (2022) and credibility prediction Leite et al. (2023) are approaches to eliminate conflicts. Approaches that are the closest to ours include prompt augmentation to improve robustness Weller et al. (2022). Despite these advances, a unified and efficient approach to handle conflicts remains a formidable challenge.

3 Problem Formulation

This paper seeks to study how a chat assistant can utilize feedback to disambiguate user questions in the setting of multi-turn user-LLM interactions.

In order to ground the notations, we consider an example of a chat system through which Human Resources can retrieve information about employees. In this situation, the conversation will be about 'entities', which are unique identifiers of the employees. Each employee will have information about them stored in relational tables and text documents. The chat system's goal is to produce natural language responses to the questions posed, by *inferring* the underlying entities. The entities may change over time as the conversation naturally evolves, or as and when an interruption arrives.

Notations

- $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$: the set of all relevant entities. In the example above, these correspond to an abstract index (such as an identification number) of each employee.
- $\mathcal{D} = \{d_1, d_2, \ldots, d_K\}$: the set of all documents referring to the entities. Each "document" is contains information regarding one or more entity. We do not assume that this information to be structured in any specific way.
- q_t, a_t, f_t: We use subscript t to indicate conversation turn. At the t-th round, q_t is the question asked by the user, a_t the answer generated by the chat-system, and f_t the feedback given the user, if any. A conversation takes the form of $\{\mathcal{D}, q_1, a_1, f_1, q_2, a_2, \ldots\}$.
- e_t^*, a_t^*: $e_t^* \in \mathcal{E}$ is the entity the user has in mind for the question q_t at round t. e_t^* is not necessarily directly expressed by the user in their questions or feedback, but to be inferred by the system through conversations. Similarly, we mark a_t^* as the true answer to question q_t, infromation from the document set $\mathcal{D}$ for the referred entity e_t^* .
- M: the large language model. We use $M(input)$ to denote the generated result.

Throughout this paper, we consider two types of user input, questions q, where the user solicits an answer, and feedbacks f, where the user expresses preference about the answer. In reality, there can be many other forms of input, but we omit them in our formulation for clarity of the system. We assume that we can separate feedbacks f_t and questions q_{t+1} even if the user input them in the same round, through, for example, a LLM parser.

Goal of the chat system. The goal of the chat system is to produce the answer after round t as $q_t(e_t^*)$, i.e., produce the answer to the question q_t based on the (unknown) latent entity e_t^*. However, the answer a_t can only depend on the available information till time t, namely all of the questions, past answers, and feedback, in addition to the document set.

4 CAN LLM EFFECTIVELY INCORPORATE FEEDBACK? AN EXPLORATION

We begin by illustrating what is lacking in the current state of the art LLMs' ability in incorporating feedback. We found that current LLMs, especially those smaller in size, fall short in tasks that involves personalization and ambiguity in 5 aspects: detecting ambiguity, persisting feedback, incorporating partial disambiguation, preference changes, and positional bias. We will illustrate these challenges respectively on a very simple synthetic dataset.

4.1 EXPERIMENT SETUP

Dataset We synthetically created short biography paragraphs of people, similar to the `Bios` dataset from Allen-Zhu & Li (2023). Each batch will consists of a few entries that share the same name, and each entry contains 5 facts: birthday, hometown, alma matar, major, current city. The questions $\mathcal{X}$ are about these five facts of this person, such as "When was Layla Gonzalez born?" or "What state is Layla Gonzalez from?". In a document batch, some facts can overlap as well, so both the name and the facts can be a source of ambiguity. For example:

> 1. Layla Gonzalez was born on February 14, 1995. They were born and raised in San Francisco, California. They went to University of Texas at Austin for higher education and majored in Art History. They currently live in Philadelphia, Pennsylvania.
> 2. Layla Gonzalez was born on August 7, 1975. They were born and raised in San Antonio, Texas. They went to UCSD for higher education and majored in Engineering. They currently live in Los Angeles, California.
> 3. Layla Gonzalez was born on February 14, 1995. They were born and raised in Phoenix, Arizona. They went to University of North Carolina at Chapel Hill for higher education and majored in Nursing. They currently live in Phoenix, Arizona.

These bios are meant to represent retrieved content for an RAG task. We constructed the synthetic data for clarity and simplicity (the bios are programmatically generated and not paraphrased as in Allen-Zhu & Li (2023)), to minimize confounding and reduce cost of these experiments.

Conversation protocol. Our experiment consists of the following protocol. Before the first question, we give the LLM all the documents serialized as a string, followed by a question. We prompt the

LLM to provide a list of plausible answers for ambiguous questions. For example, with the above set of docs and a question "When was Layla Gonzalez born?", we want the LLM to answer a list of two answers of [Feb 14 1995, Aug 7, 1975]. Subsequently, we either give the LLM feedback on the correct answer and then ask a follow-up question, or we do not provide any feedback and continue asking questions. In the case when feedback is given, we expect the LLM to be accurate in answering follow-up questions, while in the absence of feedback, we expect the LLM to return a list of answers. Conversations are in the form of $\{\mathcal{D}, x_1, a_1, f_1, x_2, a_2, f_2, \ldots\}$, as shown in Example 3.

4.2 Findings, Challenges faced

We summarize the findings from our exploratory synthetic data experiments. We use the synthetic dataset and targeted experimental design to show some of the challenges that different LLMs face for the conversation assistant task. We presetn the full metrics, experiment details, and full results in Appendix B.

Detecting ambiguity. Large models are better at detecting ambiguities. For smaller models, the ability to correctly recognize the full scope of ambiguity decreases as the number of candidates increases. (E.g. Lamma 3 70B achieves 45% precision with 3 documents, and drops to 27% for 20. See table 4) in appendix B.1 for details.

Persisting disambiguating feedback. Smaller models Llama-8B-instruct can forget feedback starting from the 3rd turn, greatly decreasing accuracy. Larger models retain feedback consistently. (Appendix B.2)

Reasoning about partial disambiguation. In the setting where ambiguous candidates are ruled out one by one through feedback, performance degrades as the reasoning chain grows longer. (Appendix B.3)

Reasoning about preference changes. We constructed a setting where the user indicates a preference change at some time in the conversation. We found that for models large or small, the performance degrades after the preference change. (Appendix B.4)

Position Bias. When multiple documents are presented to the model without disambiguation information, the model will pick the first option with overwhelming odds if asked to provide one answer only. (Appendix B.5)

Using a consistent subset of feedback improves accuracy compared to using all feedback. (Appendix B.6)

5 Personalized Feedback Coreset

5.1 Desiderata

In this section we present a novel system, CoreSet, that addresses the challenges of ambiguity by learning from user feedback. Specifically, the desiderata of the system's behavior includes:

1. Ambiguity Quantification: When there are ambiguity in a given query, the model should detect and be able to represent it.
2. Persistence: previous feedbacks should be remembered.
3. Consistency: feedbacks should not have conflict with each other.

4. Recency: prioritizes feedback closer to the current interaction.
5. Robustness: if an adversarial feedback was introduced, we should be able to recover from it.

Among them, desiderata 4 and 5 may be in conflict. We will discuss the various design trade-offs in detail in the following sections.

5.2 Background: Coreset Selection

Building a system resembling human learning abilities to sustainably learn over a long-term period is *continual learning* (Thrun, 1995). A naive continual learning method is maintaining and revisiting a small replay buffer (Rolnick et al., 2019; Titsias et al., 2019) to mitigate catastrophic forgetting. However, the majority of these methods store random-sampled or latest data as a proxy set, limiting their practicality to real-world applications when all the training data are not equally useful, as some of them can be less informative or even detrimental for the current task.

These challenges can be addressed by selecting data from the buffer by some criterion, known as *coreset* selection. A coreset refers to a small, representative subset of a large dataset that approximately preserves certain properties of the original dataset, such as its diversity or the results of a particular optimization objective. Yoon et al. (2021) showed that online coreset selection improved model performance in practical scenarios containing imbalanced, streaming, and noisy data. The method considers the diversity, task informativity, and relevancy to the past tasks.

In our context of LLM structured generation, there many concepts that are similar to a replay buffer: for example memories (Park et al., 2023b), scratchpads (Nye et al., 2021) and constitutions (Findeis et al., 2024). In this work we will explore how to select user feedback from the buffer pertaining to specific queries, and constructing a consistent and effective coreset for user queries.

5.3 Constructing the Feedback Coreset

Given a question q, with the true right answer a^*, the optimal set of feedback denoted as $\mathbf{s}^* \subseteq \mathcal{F} = \{f_1, f_2, \ldots\}$ is defined as follows.

$$\mathbf{s}^*(q, a^*) = \arg\min_{\mathbf{s} \subseteq \mathcal{F}} l(M(q, \mathbf{s}), \{a^*\})$$

Where l is a loss function that measures how much the generated answer using a feedback set $\mathbf{s} \subseteq \mathcal{F}$ denoted as $M(a, \mathbf{s})$ deviates from the ground truth answer a^*. In practice, this can be a metric such as the F1 score, or an LLM itself. For brevity, we omit the document dataset $\mathcal{D}$ in the input to M.

In reality, our algorithm does not have access to the ground truth response a^*. Therefore there is a need to find approximation of $\mathbf{s}^*$, which we will denote as $\hat{\mathbf{s}}$. There will be some loss in answering quality using the approximation $\hat{\mathbf{s}}$ as compared to the optimal feedback set $\mathbf{s}^*$, which we call *regret* for the QA pair (q, a^*). This regret given a specific (q, a^*) pair can be calculated as

$$l\big(M(q, \hat{\mathbf{s}}(q)), M(q, \mathbf{s}^*(q, a^*))\big)$$

Observe that $\hat{\mathbf{s}}$ is chosen by the algorithm and is thus only a function of q and not a^* as the true answer is not known before the set $\hat{\mathbf{s}}$ is selected.

Online Setting We formulate the user preference as a distribution over the question space and referred entities $P : \mathcal{Q} \times \mathcal{E} \to [0, 1]$. We generalize our setting to allow for user preference change by making the preference distribution subscriptable by time: $(q_t, a_t^*) \sim P_t$. We are interested in algorithms that choose $\hat{\mathbf{s}}$ as a function of q that can minimize regret on average over a QA work load, i.e., we want an algorithm that selects $\hat{\mathbf{s}}$ such that the following is minimized over a conversation horizon of T turns.

$$\sum_{t=1}^{T} \mathop{\mathbb{E}}_{(q,a^*)\sim P_t} l\big(M(q, \hat{\mathbf{s}}(q)), M(q, \mathbf{s}^*(q, a^*))\big)$$

An example of a preference change is when the latent entity changes. In particular, we can partition the time horizon into contiguous blocks of time, where in each block the latent entity is fixed, but the latent entity changes across blocks. This is commonly referred to as 'piece-wise stationary' distribution in the literature.

5.4 WHY DOES MINIMIZING REGRET IMPLY ACHIEVING THE DESIDERATA?

The loss function takes as input two sets of answers S_1 and S_2, where a member of the set is a string. Then $l(S_1, S_2)$ measures the precision and recall using the formula given before. Implementing the loss function in practice requires a subroutine that checks whether a string is in a set. In the case of structured outputs such as json, we can check for an exact match, and in the case of free-form strings, this can be done using LLMs themselves. Now we show that minimizing our loss function yields our desiderata. Let $s_i \subset \mathcal{F}$ denote the coreset we construct at the i-th turn for question q_i.

- Persistence: if a feedback f_j is useful in answering questions $q_1 \dots q_t$, then it should be in the coreset $f_j \in s_i$ for $i \in [1, \dots, t]$.
- Consistency: we show that inconsistent feedback decreases generation accuracy. Hence, $f_i \in s_i$ should not contain information that contradicts any other $f_k \in s_i$.
- Adaptivity: We seek to choose s_t be optimal with respect to P_t, rather than with respect to the past distributions $P_{t-1}, .., P_1$. In other words, when the distribution changes, that is, $P_t \neq P_{t-1}$, we want the system to optimize with respect to P_t and not with respect to the old distribution.

The optimal set $\mathbf{s}^*$ represents all the relevant information needed to answer question. For ambiguity quantification, we want the model response $M(q, \mathbf{s})$ to be "more sure", as we give more feedback.

5.5 CONSTRUCTING THE CORESET

Recency is a heuristic that is used when the underlying environments are slowly varying, i.e., in cases when $P_{t-1} \approx P_t$, we want the feedback set $\hat{\mathbf{s}}_t$ to be similar or close to that of $\hat{\mathbf{s}}_{t-1}$. For a query q, let $\mathcal{F}_q = \mathcal{F}|q \subseteq \mathcal{F}$ denote the relevant feedbacks pertaining to query q (details on retrieval in section 4.4). Let $\mathcal{D}$ denote the set of documents given as context for query q. Based on our desiderata, we have two ways of constructing the coreset $\mathbf{s}$:

Robust Coreset. Where we extract the largest consistent set of feedbacks, where adversarial or irrelevant feedbacks in memory will not be selected. This construction is closer to previous works Yoon et al. (2021) on online coreset selection.

$$\max |\mathbf{s}| \quad \text{s.t. } \mathbf{s} \text{ is consistent, } \mathcal{D}|\mathbf{s} \neq \emptyset, \ f_n \in \mathbf{s}$$

Recency Coreset Where we always treat the most recent feedback f_n as "correct" and find the largest span of previous feedback that is consistent with it.

$$\min_{1 \leq i \leq n} i \quad \text{s.t. } \mathbf{s} = \{f_i, \dots f_a\}, \ \mathbf{s} \text{ is consistent, } \mathcal{D}|\mathbf{s} \neq \emptyset$$

We provide the pseudocode for the algorithm for a Recency Coreset in Algorithm 1. It ensures that newly introduced feedback is integrated with the maximum contiguous recent subset of prior feedbacks without introducing conflicts. It allows for continuous refinement of the model's instructions while maintaining consistency with existing feedback, supporting iterative model improvement.

The system of using the constructed Corset to generate disambiguated answers is illustrated in figure 2. Note that to accomplish the pipeline we need to implement the feedback retrieval function `retrieve` and the conflict checker function `hasConflict`. The conflict checker can be naively implemented by calling a LLM and prompting it to return whether the feedbacks in $\mathbf{s}$ are consistent; or, consistency can be measured as "similarity" between the embeddings of the feedbacks. We now elaborate on how to use a RAG system for retrieval in section 4.4.

Algorithm 1 Recency CORESET

Input: All previous feedbacks M, new user query q
Output: Coreset $\mathbf{s} \subseteq M$
$\mathbf{f} \leftarrow \texttt{retrieve}(\mathcal{M}, q)$
sort $\mathbf{f} = [f_n, f_{n-1}, \ldots, f_1]$
$\mathbf{s} \leftarrow \emptyset$
for $f_i \in [f_n, f_{n-1}, \ldots, f_1]$ **do**
  if not `hasConflict`$(\mathbf{s}, f_i)$ **then**
    $\mathbf{s} \leftarrow \mathbf{s} \cup \{f_i\}$
  else
    `break` // Stop when a conflict is found
  end if
end for
return $\mathbf{s}$

Algorithm 2 Robust Recency CORESET

Input: All previous feedbacks M, new user query q
Output: Coreset $\mathbf{s} \subseteq M$
$\mathbf{f} \leftarrow \texttt{retrieve}(\mathcal{M}, q)$
sort $\mathbf{f} = [f_n, f_{n-1}, \ldots, f_1]$
$\mathbf{s} \leftarrow \emptyset$
for $f_i \in [f_n, f_{n-1}, \ldots, f_1]$ **do**
  if not `hasConflict`$(\mathbf{s}, f_i)$ **then**
    $\mathbf{s} \leftarrow \mathbf{s} \cup \{f_i\}$
  else
    `continue` // Don't add conflicting feedback to set, check previous ones
  end if
end for
return $\mathbf{s}$

5.6 IMPLEMENTATION OF HASCONFLICT

A key component of the algorithm is checking for conflicts between new feedback and the coreset, which is abstracted as the `hasConflict` function in our presentation so far. There are many approaches to implement `hasConflict`. A simple solution would be an LLM call containing the coreset $\mathbf{s}$ and the new feedback f_t, asking for a binary classification - we use this implementation in our toy experiments, and include a discussion on prompt writing in appendix D. Another approach could be comparing the distance of the embedding of $\mathbf{s}$ and f_t (e.g. cosine similarity). We say a conflict exists if the distance is below a threshold, which can be tuned online.

Because `hasConflict` is in nature a binary classifier, a natural question for extension arises: What if the classification is ambiguous, or probabilistic? In principle, one could use the feedback to tune the parameters of this classifier (such as a threshold), or use statistical techniques such as conformal prediction Angelopoulos et al. (2022) or selective classification Geifman & El-Yaniv (2017) to make guarantees to ensure that desired properties are satisfied by the final LLM output. We defer the design and analysis of tuning the classifier parameters based on textual feedback to future work.

6 EXPERIMENTS

Datasets We evaluate the proposed personalized coreset approach on (1) our `Bios` synthetic dataset and two public ambiguity benchmarks: (2) `AmbigDocs` (Lee et al., 2024) for contextual QA with ambiguous document sources. The datasets are chosen with the focus on ambiguity and are modified to a multi-turn setting.

`AmbigDocs` is constructed from Wikipedia disambiguation pages where multiple entities that can be referred to by the same surface name. Each `AmbigDocs` instance consists of a question asking about an ambiguous entity and a list of gold document-answer pairs for each disambiguated entity.

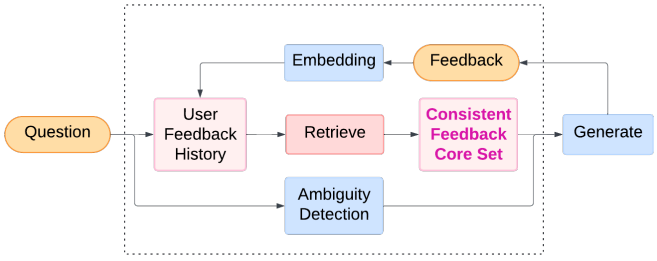


Figure 2: System design of CORESET. Orange cells are user input, Blue cells are LLM generations, and Red cells are a part of our personalization algorithm.

We generate a total of 36K examples, covering 102K unique entities over all domains. We augment the dataset into a multi-turn disambiguation dataset `DisAmbigDocs` by repeating the process for question-answer pair generation, leveraging Claude Sonnet 3.5. See figure 7 for an illustration of the generation process. This is done by providing the documents and existing question to the LLM and asking it to create a different question. We then verify the correctness and quality of the answer.

Baselines, models, and metrics The baseline methods are categorized into two groups: (a) generic reasoning, which includes QA with in-context learning (ICL), Chain-of-Thought (Wei et al., 2022) (CoT); and (b) Memory methods, which includes unfiltered memory, scratchpads (Nye et al., 2021), and summarized memory (similar approach in Park et al. (2023b)).

We will continue to use top-K precision and recall as defined in section 4 and appendix B.1. In our experiments, we use two LLaMA models by Dubey et al. (2024) (llama3-8b-instruct, llama3-70b-instruct), and Claude 3 Sonnet. In the main paper, we omit the llama3-8b-instruct results as they do not perform as well as llama3-70b-instruct. Examples of these prompts can be found in the Appendix. Details regarding the LLM inference parameters and the number of demonstration samples used are provided in the Appendix.

6.1 RESULTS

Table 2 presents the comparative performance of multiple baseline models across two distinct disambiguation interactions. This analysis aims to highlight the effectiveness of each approach in addressing the challenges posed by disambiguation tasks.

We see that our method CoreSet algorithm mostly outperforms all baselines, showing stronger disambiguation and persistence performance, especially for the stronger Claude-3-Sonnet model. One surprising finding is the low performance of Scratchpads, whose mechanism is similar to our method in concept. A reason for this performance gap is due to conflicting information in the generated scratchpads due to preference changes (see figure 8 for example).

Table 3 shows the number of average tokens for the 50-turn conversation experiment, to highlight the computational advantage of CoreSet compared to other well-performing methods such as full memory or summarized memory.

	Method	Bios		AmbigDocs	
		Precision	Recall	Precision	Recall
Llama-3-70B	**Reasoning Methods**				
	ICL	0.69	0.72	0.65	**0.71**
	CoT	0.36	0.68	0.34	0.42
	Memory Methods				
	full memory	0.72	0.76	0.42	0.39
	Scratchpads	0.16	0.29	0.12	0.23
	Summarized Memory	0.72	0.77	0.44	0.57
	CoreSet	**0.79**	**0.83**	**0.66**	0.60
Claude-3-Sonnet	**Reasoning Methods**				
	ICL	0.85	0.90	0.53	0.69
	CoT	0.49	0.60	0.44	0.41
	Memory Methods				
	full memory	0.94	0.95	0.60	0.70
	Scratchpads	0.39	0.65	0.31	0.59
	Summarized Memory	0.89	0.92	0.58	0.70
	CoreSet	**0.95**	**0.98**	**0.62**	**0.70**

Table 2: Performance comparison across two Datasets: Bios and AmbigDocs. The best results are highlighted in bold.

	Method	Bios		AmbigDocs	
		# in	# out	# in	# out
Llama-3-70B	**Reasoning Methods**				
	ICL	903	1274	1697	3277
	CoT	936	10831	1185	36892
	Memory Methods				
	unfiltered memory	747	1143	863	3234
	Scratchpads	32755	9302	39132	10532
	Summarized Memory	5944	7936	7827	8314
	CoreSet	3736	7122	3652	8127

Table 3: Comparison of computational cost for each baseline. We report the average number of input and output tokens in one 50-question conversation. Our method is more efficient than memory methods.

7 DISCUSSION

In this work, we addressed the challenges of ambiguity and personalized feedback in LLMs by proposing a novel system to dynamically construct a core set of relevant feedback, with a future work extension leveraging Retrieval-Augmented Generation (RAG) explored in Appendix A. We identified key issues that current LLMs face, such as inconsistent feedback utilization, difficulties

in reasoning with multiple feedbacks, and adapting to evolving user preferences. To tackle these problems, we introduced the CORESET algorithm, which effectively selects the most contextually relevant feedback without conflicts, enhancing the model's ability to persist user preferences over time. Our approach demonstrated significant improvements over state-of-the-art methods on synthetic, contextual QA, and text-to-SQL benchmarks, showing that incorporating structured, personalized feedback can substantially enhance both the accuracy and robustness of LLMs. These results highlight the importance of personalization and adaptability in LLMs, paving the way for more intelligent, user-centric AI systems that can better handle real-world ambiguity and evolving user needs.

The limitation of our work is that we focus on a narrow scenario, where the model has access to retrieved documents, source of ambiguity is limited to denotation, and that it receives ample user feedback. Future work may extend this approach to more diverse forms of use cases and sources of ambiguities.

REFERENCES

Ahdritz, G., Qin, T., Vyas, N., Barak, B., and Edelman, B. L. Distinguishing the knowable from the unknowable with language models. *arXiv preprint arXiv:2402.03563*, 2024.

Allen-Zhu, Z. and Li, Y. Physics of language models: Part 3.2, knowledge manipulation. *arXiv preprint arXiv:2309.14402*, 2023.

Amplayo, R. K., Webster, K., Collins, M., Das, D., and Narayan, S. Query refinement prompts for closed-book long-form question answering. *arXiv preprint arXiv:2210.17525*, 2022.

An, S., Ma, Z., Lin, Z., Zheng, N., and Lou, J.-G. Make your llm fully utilize the context. *arXiv preprint arXiv:2404.16811*, 2024.

Angelopoulos, A. N., Bates, S., Fisch, A., Lei, L., and Schuster, T. Conformal risk control. *arXiv preprint arXiv:2208.02814*, 2022.

Bhaskar, A., Tomar, T., Sathe, A., and Sarawagi, S. Benchmarking and improving text-to-sql generation under ambiguity. *arXiv preprint arXiv:2310.13659*, 2023.

Chen, A., Gudipati, P., Longpre, S., Ling, X., and Singh, S. Evaluating entity disambiguation and the role of popularity in retrieval-based nlp. *arXiv preprint arXiv:2106.06830*, 2021.

Cole, J. R., Zhang, M. J., Gillick, D., Eisenschlos, J. M., Dhingra, B., and Eisenstein, J. Selectively answering ambiguous questions. *arXiv preprint arXiv:2305.14613*, 2023.

Desai, S. and Durrett, G. Calibration of pre-trained transformers. *arXiv preprint arXiv:2003.07892*, 2020.

Du, Y., Bosselut, A., and Manning, C. D. Synthetic disinformation attacks on automated fact verification systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 10581–10589, 2022.

Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Yang, A., Fan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Findeis, A., Kaufmann, T., Hüllermeier, E., Albanie, S., and Mullins, R. Inverse constitutional ai: Compressing preferences into principles. *arXiv preprint arXiv:2406.06560*, 2024.

Gao, G., Taymanov, A., Salinas, E., Mineiro, P., and Misra, D. Aligning llm agents by learning latent preference from user edits. *arXiv preprint arXiv:2404.15269*, 2024.

Geifman, Y. and El-Yaniv, R. Selective classification for deep neural networks. *Advances in neural information processing systems*, 30, 2017.

Geng, J., Cai, F., Wang, Y., Koeppl, H., Nakov, P., and Gurevych, I. A survey of confidence estimation and calibration in large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6577–6595, 2024.

Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Duan, N., and Chen, W. Critic: Large language models can self-correct with tool-interactive critiquing. *arXiv preprint arXiv:2305.11738*, 2023.

He, J., Pan, K., Dong, X., Song, Z., LiuYiBo, L., Qianguosun, Q., Liang, Y., Wang, H., Zhang, E., and Zhang, J. Never lost in the middle: Mastering long-context question answering with position-agnostic decompositional training. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13628–13642, 2024.

Hou, B., Liu, Y., Qian, K., Andreas, J., Chang, S., and Zhang, Y. Decomposing uncertainty for large language models through input clarification ensembling. *arXiv preprint arXiv:2311.08718*, 2023.

Kim, G., Kim, S., Jeon, B., Park, J., and Kang, J. Tree of clarifications: Answering ambiguous questions with retrieval-augmented large language models. *arXiv preprint arXiv:2310.14696*, 2023.

Köpf, A., Kilcher, Y., von Rütte, D., Anagnostidis, S., Tam, Z. R., Stevens, K., Barhoum, A., Nguyen, D., Stanley, O., Nagyfi, R., et al. Openassistant conversations-democratizing large language model alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

Lee, D., Kim, S., Lee, M., Lee, H., Park, J., Lee, S.-W., and Jung, K. Asking clarification questions to handle ambiguity in open-domain qa. *arXiv preprint arXiv:2305.13808*, 2023.

Lee, Y., Ye, X., and Choi, E. Ambigdocs: Reasoning across documents on different entities under the same name. *arXiv preprint arXiv:2404.12447*, 2024.

Leite, J. A., Razuvayevskaya, O., Bontcheva, K., and Scarton, C. Detecting misinformation with llm-predicted credibility signals and weak supervision. *arXiv preprint arXiv:2309.07601*, 2023.

Li, Y., Wen, H., Wang, W., Li, X., Yuan, Y., Liu, G., Liu, J., Xu, W., Wang, X., Sun, Y., et al. Personal llm agents: Insights and survey about the capability, efficiency and security. *arXiv preprint arXiv:2401.05459*, 2024.

Liu, A., Wu, Z., Michael, J., Suhr, A., West, P., Koller, A., Swayamdipta, S., Smith, N. A., and Choi, Y. We're afraid language models aren't modeling ambiguity. *arXiv preprint arXiv:2304.14399*, 2023.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Manakul, P., Liusie, A., and Gales, M. J. Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. *arXiv preprint arXiv:2303.08896*, 2023.

Min, S., Michael, J., Hajishirzi, H., and Zettlemoyer, L. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*, 2020.

Nye, M., Andreassen, A. J., Gur-Ari, G., Michalewski, H., Austin, J., Bieber, D., Dohan, D., Lewkowycz, A., Bosma, M., Luan, D., et al. Show your work: Scratchpads for intermediate computation with language models. *arXiv preprint arXiv:2112.00114*, 2021.

Park, J., Lim, S., Lee, J., Park, S., Chang, M., Yu, Y., and Choi, S. Clara: classifying and disambiguating user commands for reliable interactive robotic agents. *IEEE Robotics and Automation Letters*, 2023a.

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., and Bernstein, M. S. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023b.

Piantadosi, S. T., Tily, H., and Gibson, E. The communicative function of ambiguity in language. *Cognition*, 122(3):280–291, 2012.

Pilault, J., Garcia, X., Bražinskas, A., and Firat, O. Interactive-chain-prompting: Ambiguity resolution for crosslingual conditional generation with interaction. *arXiv preprint arXiv:2301.10309*, 2023.

Quach, V., Fisch, A., Schuster, T., Yala, A., Sohn, J. H., Jaakkola, T. S., and Barzilay, R. Conformal language modeling. *arXiv preprint arXiv:2306.10193*, 2023.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., and Wayne, G. Experience replay for continual learning. *Advances in neural information processing systems*, 32, 2019.

Ross, S. I., Martinez, F., Houde, S., Muller, M., and Weisz, J. D. The programmer's assistant: Conversational interaction with a large language model for software development. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*, pp. 491–514, 2023.

Shrivastava, V., Liang, P., and Kumar, A. Llamas know what gpts don't show: Surrogate models for confidence estimation. *arXiv preprint arXiv:2311.08877*, 2023.

Si, C., Gan, Z., Yang, Z., Wang, S., Wang, J., Boyd-Graber, J., and Wang, L. Prompting gpt-3 to be reliable. *arXiv preprint arXiv:2210.09150*, 2022.

Sun, W., Cai, H., Chen, H., Ren, P., Chen, Z., de Rijke, M., and Ren, Z. Answering ambiguous questions via iterative prompting. *arXiv preprint arXiv:2307.03897*, 2023.

Thrun, S. A lifelong learning perspective for mobile robot control. In *Intelligent robots and systems*, pp. 201–214. Elsevier, 1995.

Titsias, M. K., Schwarz, J., Matthews, A. G. d. G., Pascanu, R., and Teh, Y. W. Functional regularisation for continual learning with gaussian processes. *arXiv preprint arXiv:1901.11356*, 2019.

Tonolini, F., Massiah, J., Aletras, N., and Kazai, G. Bayesian prompt ensembles: Model uncertainty estimation for black-box large language models. 2024.

Vasconcelos, H., Bansal, G., Fourney, A., Liao, Q. V., and Vaughan, J. W. Generation probabilities are not enough: Exploring the effectiveness of uncertainty highlighting in ai-powered code completions. *arXiv preprint arXiv:2302.07248*, 2023.

Wang, X., Wang, Z., Liu, J., Chen, Y., Yuan, L., Peng, H., and Ji, H. Mint: Evaluating llms in multi-turn interaction with tools and language feedback. *arXiv preprint arXiv:2309.10691*, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Weller, O., Khan, A., Weir, N., Lawrie, D., and Van Durme, B. Defending against misinformation attacks in open-domain question answering. *arXiv preprint arXiv:2212.10002*, 2022.

Xu, R., Qi, Z., Wang, C., Wang, H., Zhang, Y., and Xu, W. Knowledge conflicts for llms: A survey. *arXiv preprint arXiv:2403.08319*, 2024.

Yoon, J., Madaan, D., Yang, E., and Hwang, S. J. Online coreset selection for rehearsal-based continual learning. *arXiv preprint arXiv:2106.01085*, 2021.

Zhang, J., Li, Z., Das, K., Malin, B. A., and Kumar, S. Sac: Reliable hallucination detection in black-box language models via semantic-aware cross-check consistency. *arXiv preprint arXiv:2311.01740*, 2023.

Zhang, M. J. and Choi, E. Situatedqa: Incorporating extra-linguistic contexts into qa. *arXiv preprint arXiv:2109.06157*, 2021.

Zhang, M. J. and Choi, E. Clarify when necessary: Resolving ambiguity through interaction with lms. *arXiv preprint arXiv:2311.09469*, 2023.

Zhang, Z., Yang, F., Jiang, Z., Chen, Z., Zhao, Z., Ma, C., Zhao, L., and Liu, Y. Position-aware parameter efficient fine-tuning approach for reducing positional bias in llms. *arXiv preprint arXiv:2404.01430*, 2024.

Zhong, W., Guo, L., Gao, Q., Ye, H., and Wang, Y. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19724–19731, 2024.

A FUTURE WORK: EXTRACTING FEEDBACK USING RAG

So far, we have assumed that the feedbacks users provided are explicit, accurate, and relevant to the questions. In real-world conversations, however, these conditions might not be perfectly met. Therefore, it would be necessary to extract accurate and relevant feedback to a question from past interaction history through retrieval.

Retrieval-Augmented Generation (RAG) leverages dense embeddings to retrieve relevant past memories from a datastore to enhance the response generation process for a given user query. Given a user query q, an embedding $\mathbf{e}_q \in \mathbb{R}^d$ is computed using a pretrained encoder $E(\cdot)$. The datastore consists of a set of memory entries $M = \{f_1, f_2, \ldots, f_N\}$, where each memory f_i is represented by its dense embedding $\mathbf{e}_{f_i} \in \mathbb{R}^d$. To retrieve relevant memories, the cosine similarity between $\mathbf{e}_q$ and each $\mathbf{e}_{f_i}$ is computed as:

$$\text{sim}(\mathbf{e}_q, \mathbf{e}_{f_i}) = \frac{\mathbf{e}_q \cdot \mathbf{e}_{f_i}}{\|\mathbf{e}_q\| \|\mathbf{e}_{f_i}\|}$$

The top-k feedbacks with the highest similarity scores are selected as the relevant set $R \subset M$. These retrieved conversations can then be used as feedback input to our CoreSet algorithms.

B CAN LLM EFFECTIVELY INCORPORATE FEEDBACK? AN EXPLORATION

Performance Metrics. In tasks involving ambiguity, the algorithm will often return a set of answers instead of only one. Following previous works in information retrieval and recommendation systems, we use Top-k precision and recall to evaluate both the accuracy and the uncertainty quantification of the model. Let $\mathbf{z} \in \mathcal{E}$ be a set of entities returned by the algorithm, and $\mathbf{z}^* \in \mathcal{E}$ be the ground truth set of relevant entities given previous feedbacks. If the previous feedbacks contain enough information to disambiguate denotation, then $\mathbf{z}^* = e^*$.

$$\text{Precision} = \frac{\text{Number of relevant items in set}}{\text{Size of set}} = \frac{\sum_{e \in \mathbf{z}} \mathbb{I}(e \in \mathbf{z}^*)}{|\mathbf{z}|}$$

$$\text{Recall} = \frac{\text{Number of relevant items in set}}{\text{Total number of relevant items}} = \frac{\sum_{e \in \mathbf{z}} \mathbb{I}(e \in \mathbf{z}^*)}{|\mathbf{z}^*|}$$

Notations We formulate our task as follows.

- $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$: relevant entities for the task.
- $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$: documents, each for each entity
- $\mathcal{X}$: the space of questions one can ask about the entities. We write $x \in \mathcal{X}$ as the question, and $x(e_1)$ as the answer of question x for entity 1.
- z: a positive integer, indicating the user's preferred entity
- t: subscript to indicate turn. The questions will be indexed as x_t, the answers a_t, and the feedbacks f_t. A conversation, for example, can take the form of $\{\mathcal{D}, x_1, a_1, f_1, x_2, a_2, \ldots\}$
- F: The feedback function. We define the feedback as a function of the true denotation and past conversations, i.e. $f_1 = F(z_t, x_{1:t}, a_{1:t}, f_{1:t-1})$

B.1 CHALLENGE 0: DETECTING AMBIGUITY

We first look at the case without feedback. The prompt consists of the ambiguous documents, and we ask the model k questions in conversation: $\{\mathcal{D}, x_1, a_1\}$ (see Example 1). The desired behaviour is that the model returns a list of answers, one for each entity in the given set $\{x_1(e_1), x_1(e_2), \ldots, x_1(e_N)\}$.

The problem of identifying ambiguity has been widely studied with approaches ranging from directly asking the LLM if it is ambiguous (Zhang & Choi, 2023) to prompting the model multiple times and evaluating the diversity of answers (Hou et al., 2023). Also see Lee et al. (2024) for a detailed discussion.

Model	3 docs		10 docs		20 docs	
	Precision	Recall	Precision	Recall	Precision	Recall
Llama 3 8B	0.21	0.66	0.18	0.39	0.21	0.35
Llama 3 70B	0.45	0.76	0.26	0.45	0.27	0.41
Claude 3 Sonnet	1.0	1.0	1.0	1.0	1.0	1.0

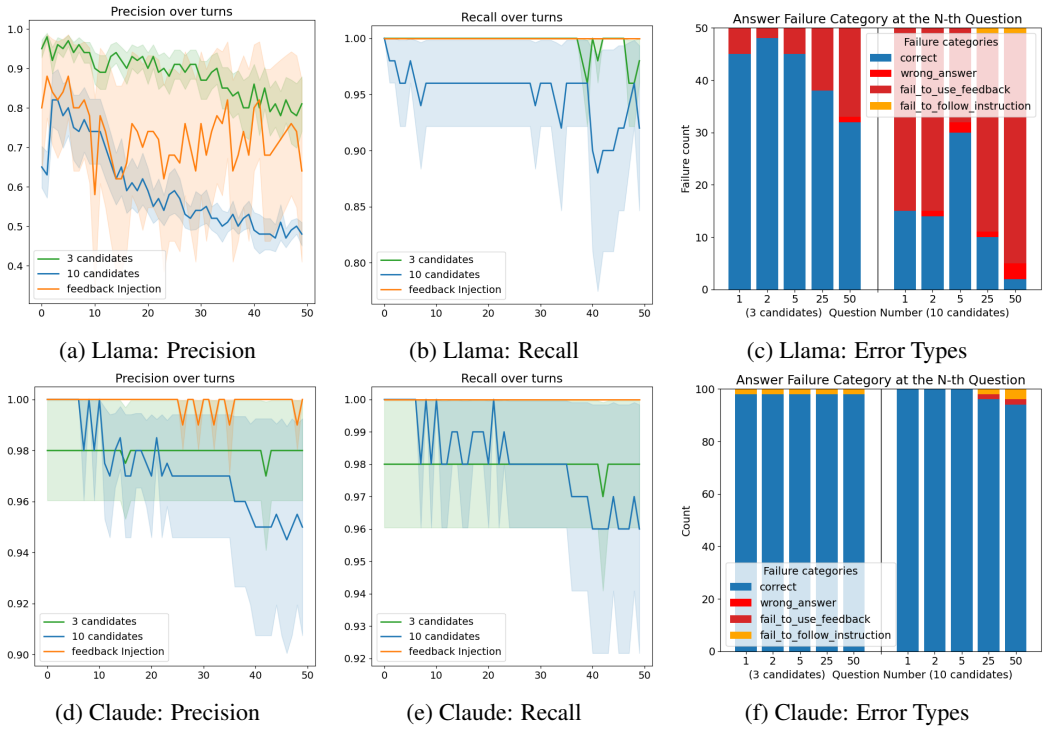
Table 4: Representing ambiguity

From the results in table 4), we can see the following: Large models are better at detecting ambiguities. For smaller models, the ability to correctly recognize the full scope of ambiguity decreases as the number of candidates increases. (E.g. Lamma 3 70B achieves 45% precision with 3 documents, and drops to 27% for 20.

B.2 Challenge 1: Persistence

In this task, we introduce feedback after one turn of conversation $\{\mathcal{D}, x_1, a_1, f_1\}$, in the form of providing the correct answer to x_1: $f_1 = x_1(z)$ in natural language. Afterwards, for each turn $t > 1$, we randomly choose a question from $\mathcal{X}$ (which includes x_1), and evaluate the answers. We provide an example conversation in the appendix in Example 2. Every time we ask a new question, all previous conversation history is included in the prompt.

Figure 3 and table 6 illustrate how the accuracy decreases over time for as we ask questions. This behaviour of deteriorating persistence is exacerbated when the model is presented with more ambiguous candidates (comparing the results for $N = 3$ candidate documents and $N = 10$ candidate documents.



(a) Llama: Precision (b) Llama: Recall (c) Llama: Error Types

(d) Claude: Precision (e) Claude: Recall (f) Claude: Error Types

Figure 3: Precision, recall, and error types over turn t for challenge 1. There is sharp decline in the model's ability to persist the instruction and feedback after a few turns.

Turn	1		5		25		50	
	Precision	Recall	Prec	Rec	Prec	Rec	Prec	Rec
3 documents								
Claude-sonnet	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
Llama3 70B	0.95	1.0	0.95	1.0	0.88	1.0	0.81	0.98
10 documents								
Claude-sonnet	1.0	1.0	1.0	1.0	0.97	0.98	0.95	0.96
Llama3 70B	0.65	1.0	0.78	0.96	0.58	0.96	0.48	0.92

Table 5: The behaviour of deteriorating persistence, especially when exacerbated with more ambiguous candidates, is evident across models.

B.3 Challenge 2: Incorporating partial disambiguation

We further expand the setting of challenge 1 to the case when one feedback does not fully disambiguate the answer. We introduce the feedbacks sequentially, in the form of $\{\mathcal{D}, x_1, a_1, f_1, x_2, a_2, f_2, \ldots\}$, as shown in Example 3.

In this setting, we do not evaluate against the ground truth z^* because there are not enough information provided. Instead, we will use $\mathbf{z}_t^*$ to denote the set of relevant entities given all previous feedbacks $f_{1:t}$. The key difference from challenge 1 is that the size of $\mathbf{z}_t^*$ can be greater than one. We designed the feedbacks to be always useful in narrowing down the set, i.e. $\cdots \subset \mathbf{z}_2^* \subset \mathbf{z}_1^* \subset \mathcal{E}$. In the following experiment, we assume each feedback reduces the size of $\mathbf{z}_t^*$ by 1.

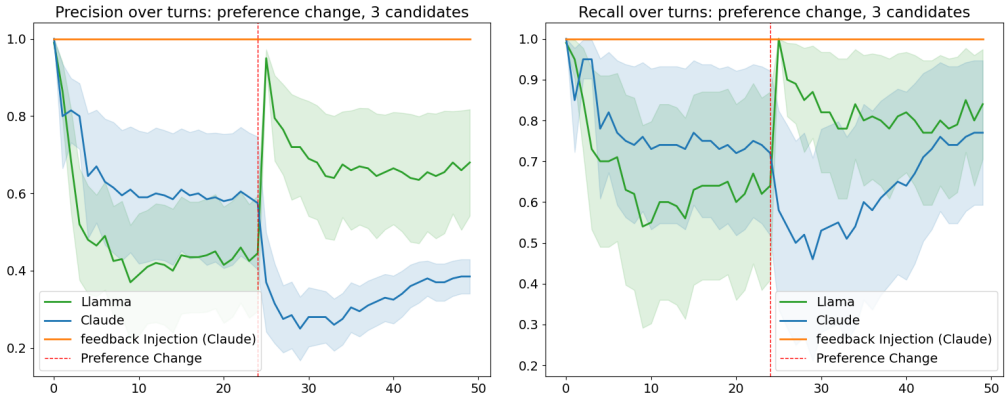
Feedbacks	0		1		2		3	
	Precision	Recall	Prec	Rec	Prec	Rec	Prec	Rec
3 documents								
Claude 3 sonnet	1.00	1.00	1.00	0.92	1.00	1.00	/	/
Llama 3 70B	0.90	0.90	0.88	0.87	0.89	0.91	/	/
10 documents								
Claude 3 sonnet	1.00	0.99	0.98	0.80	0.99	0.62	0.90	0.41
Llama 3 70B	0.95	0.95	0.81	0.88	0.80	0.92	0.73	0.96

Table 6: The behaviour of deteriorating persistence, especially when exacerbated with more ambiguous candidates, is evident across models.

B.4 Challenge 3: Preference Changes

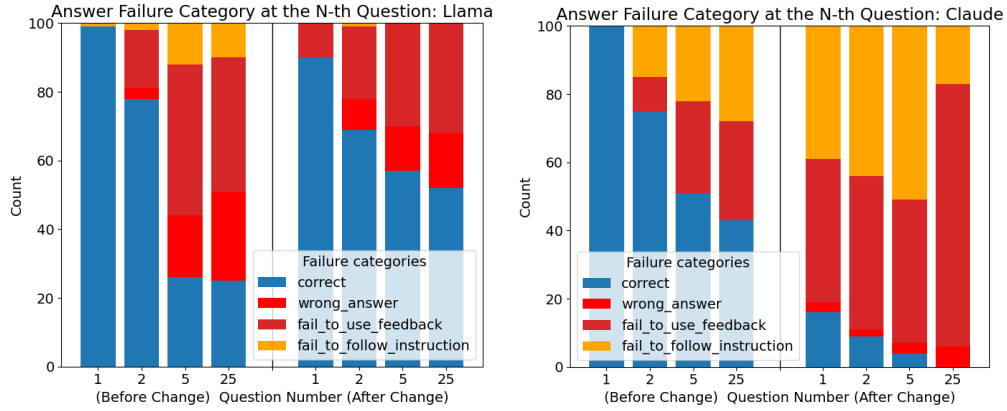
For the third challenge, we study the effect of relaxing the assumption of the denotation being consistent over time: instead of a single z^*, there can be a time step t_c where the user's preference changes, i.e. $z^*_{t_{c-1}} \neq z^*_{t_c}$. We present result in Figure 4 and analysize the error type in 5. In these experiments, the denoted entity switches to a different random document at $t = 25$. The change is conveyed through user feedback, similar to the exchange shown in figure 1.

We observe different behavioral patterns exhibited by the Llama and Claude models. For Llamma 3 70B, the new feedback from preference change reminds the model of the instruction, and improves performance temporarily. For Claude, however, the feedback further decreases the accuracy of returned answers.



(a) Precision (b) Recall

Figure 4: Precision and recall over turn t for challenge 3 preference change experiments.



(a) Llama: Error types (b) Claude: Error types

Figure 5: Error types over turn t for challenge 3 preference change experiments.

B.5 Challenge 4: Positional Bias

Positional bias is a well-known behavior found in LLMs, where when presented with a long context, the models fail to utilize information in the middle of the context (Liu et al., 2024; An et al., 2024). In this experiment, the setting is the same as Challenge 2, where multiple documents are presented to the model without disambiguation information. Instead of reporting the entire ambiguous set, we ask the model to pick an answer. Figure 6 shows our finding of models preferring the first option over others consistently. Positional bias is a motivation for uncertainty quantification, because without specific prompting the models will ignore information present in the context.

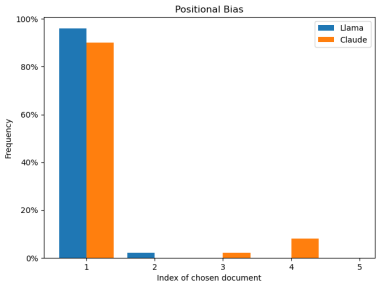


Figure 6: Positional preference.

B.6 Using a subset of the feedback improves accuracy compared to using all feedback

Convention tells us that throwing all the information to the LLM and let it figure out the useful ones. However, LLMs inherently lack an explicit conception of temporal dynamics. Consequently, when inundated with conflicting feedback within a single prompt, there is no guarantee that the model will prioritize or adhere to the most recent directive a notion of time, throwing in conflicting feedback

into the prompt will not automatically result in the LLM following the latest prompt. Even adding in an instruction to follow the latest feedback, and then adding a numbering scheme to each feedback is in-sufficient as seen in the prior section. Further, always adding all feedback into the prompt makes inference cost prohibitive.

Consider for instance a conversation history, where the interactions are as follows.

```
User: Where was Maya born?

Assistant: There are two different persons with names Maya - Maya
Jones and James Maya.

User (Feedback): The one who is the younger of the two.

Assistant: James Maya was born in San Francisco.

User (feedback): What about the other Maya?

Assistant: Maya Jones was born in Seattle.

User: What is their education level?
```

Here, using all the conversation history in the prompt is not ideal as there is conflicting information on which Maya (the entity) is being referred to in the last question.

Feedbacks	All feedbacks		Selected recent relevant feedbacks	
	Precision	Recall	Precision	Recall
3 documents				
Claude 3 sonnet	0.45	0.78	1.00	0.99
Llama 3 70B	0.39	0.54	0.92	0.99
10 documents				
Claude 3 sonnet	0.33	0.79	0.98	1.00
Llama 3 70B	0.21	0.45	0.90	0.91

Table 7: Experiment results showing model behavior at $t = 30$ after 3 preference changes. The results show that using a subset of the feedback improves accuracy compared to using all feedback.

C Dataset Generation

An illustration of our data augmentation process of the AmbigDocs benchmark can be found in figure 7.

D Prompt and interaction Examples

D.1 Experiment-related Examples

We include an example showing sample stratchpad and coreset output of the same interaction in 8.

D.2 Challenge 0: Ambiguity

Example 1 (Ambiguity)**.**

```
Here are a list of biographies in our database.

Layla Gonzalez was born on October 9, 1967. They were born and
raised in San Diego, California. They went to University of Texas
at Austin for higher education and majored in Art History. They
currently live in Philadelphia, Pennsylvania.
-
Layla Gonzalez was born on August 7, 1975. They were born and
raised in San Antonio, Texas. They went to UCSD for higher
education and majored in Engineering. They currently live in Los
```

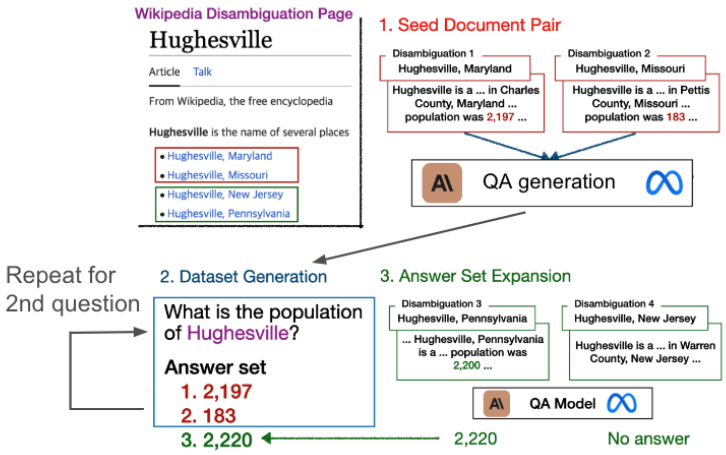


Figure 7: Generation illustration for the AmbigDocs dataset used in our experiments, modified from the original one from Lee et al. (2024).

```
Ryan Rivera was born on October 17, 1973. They were born and raised in Denver, Colorado. They went to Harvard University
for college. Durring college, they majored in Philosophy. They currently live in Washington, D.C..
Ryan Rivera was born on March 5, 1980. They were born and raised in Charlotte, North Carolina. They went to University of
Virginia for college. Durring college, they majored in Philosophy. They currently live in Houston, Texas.

Ryan Rivera was born on October 17, 1973. They were born and raised in Denver, Colorado. They went to Harvard University
for college. Durring college, they majored in Philosophy. They currently live in Washington, D.C..
Ryan Rivera was born on March 5, 1980. They were born and raised in Charlotte, North Carolina. They went to University of
Virginia for college. Durring college, they majored in Philosophy. They currently live in Houston, Texas.

Ryan Rivera was born on October 17, 1973. They were born and raised in Denver, Colorado. They went to Harvard University
for college. Durring college, they majored in Philosophy. They currently live in Washington, D.C..
Ryan Rivera was born on June 18, 1998. They were born and raised in Indianapolis, Indiana. They went to Princeton Univers
ity for college. Durring college, they majored in Biology. They currently live in Fort Worth, Texas.
Ryan Rivera was born on June 18, 1998. They were born and raised in Indianapolis, Indiana. They went to Princeton Univers
ity for college. Durring college, they majored in Biology. They currently live in Fort Worth, Texas.
```

(a) Scratchpads

```
- Ryan Rivera currently lives in Boston, Massachusetts.
- There is one biography that mentions Ryan Rivera living in Boston, Massachusetts and majoring in History:
    Ryan Rivera was born on March 11, 2005. They were born and raised in Dallas, Texas. They went to Stanford University
for college. Durring college, they majored in History. They currently live in Boston, Massachusetts.
```

(b) Coreset

Figure 8: An example of Scratchpads vs. CoreSet at conversation turn 25. The performance improvement is due to the consistency property of CoreSet, which eliminates conflicts in the notes provided back to the chat system.

```
Angeles, California.
_
Layla Gonzalez was born on February 14, 1995.  They were born
and raised in Phoenix, Arizona.  They went to University of
North Carolina at Chapel Hill for higher education and majored
in Nursing.  They currently live in Phoenix, Arizona.

Answer the user queries with the answer only.  When you are not
confident, answer a list of responses, one response per row.

User:  when was Layla Gonzalez born?

Answer:  October 9, 1967
August 7, 1975
February 14, 1995
```

D.3 CHALLENGE 1: PERSISTENCE

Example 2 (Persistence). Here are a list of biographies in our database.

Layla Gonzalez was born on October 9, 1967. They were born and raised in San Diego, California. They went to University of Texas at Austin for higher education and majored in Art History. They currently live in Philadelphia, Pennsylvania.
–
Layla Gonzalez was born on August 7, 1975. They were born and raised in San Antonio, Texas. They went to UCSD for higher education and majored in Engineering. They currently live in Los Angeles, California.
–
Layla Gonzalez was born on February 14, 1995. They were born and raised in Phoenix, Arizona. They went to University of North Carolina at Chapel Hill for higher education and majored in Nursing. They currently live in Phoenix, Arizona.

Answer the user queries with the answer only. When you are not confident, answer a list of responses, one response per row.

User: when was Layla Gonzalez born?

Answer: October 9, 1967
August 7, 1975
February 14, 1995

User: Layla Gonzalez was born on August 7, 1975. Use this information for your answer.

D.4 Challenge 2: Partial Disambiguation

Example 3 (Partial Disambiguation). Here are a list of biographies in our database.

Mia Ramirez was born on February 2, 2000. They were born and raised in San Diego, California. They went to Cornell University for higher education and majored in Anthropology. They currently live in San Diego, California.
–
Mia Ramirez was born on April 27, 1993. They were born and raised in Washington, D.C.. They went to Yale University for higher education and majored in Journalism. They currently live in Austin, Texas.
–
Mia Ramirez was born on April 27, 1993. They were born and raised in Seattle, Washington. They went to Massachusetts Institute of Technology for higher education and majored in Nursing. They currently live in Nashville, Tennessee.

Answer the user questions. When there is only one answer, return the answer plainly, no extra information. When there are multiple answers, provide the full list of answers, one answer per row.

User: Which university did Mia Ramirez attend?

Answer: Cornell University
Yale University
Massachusetts Institute of Technology

User: Mia Ramirez was born on April 27, 1993. Use this information for your answers.

User: Where does Mia Ramirez live?

Answer: Austin, Texas
Nashville, Tennessee

User: Mia Ramirez is from Seattle, Washington. Use this information for your answers.

User: What subject did Mia Ramirez study?

Answer: Nursing