Sherlock Your Queries: Learning to Ask the Right Questions for Dialogue-Based Retrieval

Dong Yun, Marco Schouten, Dim Papadopoulos Technical University of Denmark (DTU) Anker Engelunds Vej 101, 2800 Kongens Lyngby

{s232293, marschou, dimp}@dtu.dk

Abstract

User queries in information retrieval are often ambiguous, making it challenging for systems to identify a user's target from a single query. While recent dialogue-based interactive retrieval systems can clarify user intent, they are inefficient as they often lack an explicit strategy to ask the most informative questions. To address this limitation, we propose SherlockLLM, a dialogue-driven retrieval framework that learns an optimal questioning strategy via Reinforcement Learning (RL) and avoids the need for large-scale annotated dialogue data. In our framework, an agent is trained to generate a sequence of binary questions to efficiently narrow down the search space. To validate our approach, we introduce a benchmark with both structured and unstructured tasks. Experimental results show that SherlockLLM is a robust and efficient solution. On the structured tasks, its performance matches strong baselines and approaches the theoretical optimal defined by binary search. On the challenging unstructured task, our agent significantly outperforms these baselines, showcasing its ability to learn a highly effective information-seeking dialogue policy. We will release the code and models upon acceptance.

1. Introduction

User queries in information retrieval tasks are often ambiguous or incomplete [49]. This poses a significant challenge for traditional retrieval systems that rely on precise and comprehensive input. This is particularly evident in scenarios where users have a clear target in mind but lack the specific keywords to describe it, such as trying to find an image of a celebrity whose name they cannot recall. In such cases, a simple, one-shot query is often insufficient to guarantee the retrieval of a specific target image.

Dialogue-based interactive retrieval offers an effective solution by enabling systems to clarify user intent through



Figure 1. Our dialogue-based information retrieval agent. Without precise target queries, the agent identifies the desired target through a series of yes/no questions. At each turn, the agent generates the most informative question for the user, minimizing the number of interactions needed to determine the target.

multi-turn interactions [45, 48]. As illustrated in Fig. 1, an agent can engage in a conversation, asking targeted ques-

^{*}These authors contributed equally.

tions to iteratively narrow down a large pool of candidates and pinpoint the correct item. A well-formulated question has been shown to improve retrieval efficiency by as much as 100% [2]. Recent approaches have used Large Language Models (LLMs) to generate clarification questions in retrieval systems [20, 21, 23, 40]. However, the reasoning capabilities of LLMs do not inherently translate into effective information-seeking strategies. This is because the models lack an explicit learning signal aligned with the objective of uncertainty reduction [13]. Therefore, the central challenge is to generate the most informative question to minimize the number of iterations required at each dialogue turn.

In this work, we propose *SherlockLLM*, a dialogue-based retrieval framework that learns this strategic questioning ability. We formulate the task as a sequential decision-making problem solved via RL. The objective is to learn a policy that generates a sequence of questions to minimize the number of turns needed to achieve a satisfactory retrieval outcome. We train an agent to interact with a retrieval environment, where it receives rewards based on how effectively its generated questions reduce the candidate space. Through this process, the agent autonomously learns a dialogue policy optimized for retrieval efficiency.

To validate SherlockLLM's effectiveness and generalizability, we evaluate it on two distinct retrieval tasks with both structured and unstructured data. For structured data, we introduce a benchmark composed of two tasks: Guess Number and Guess Who. These tasks provide a controlled environment for evaluating the agent's ability to filter and retrieve targets via dialogue. The agent is guided by a reward function based on Expected Information Gain (EIG), and evaluates retrieval efficiency using binary search as a reference for optimal performance. For the more complex task of unstructured image retrieval, the agent's questioning policy is guided by image captions. Here, the reward function is based on the rank of the target image in a text-based search, directly optimizing for retrieval performance.

Our key contributions are threefold:

- 1. We propose *SherlockLLM*, a RL framework for dialogue-driven retrieval that autonomously learns a policy to generate optimally informative questions.
- 2. We introduce a benchmark comprising both structured (Guess Number and Guess Who) and unstructured (Image Retrieval) tasks to assess dialogue policy.
- 3. Experimental results demonstrate that our approach is a robust and generalizable solution for interactive information retrieval. On tabular tasks, its performance matches strong baselines like DeepSeek-V3.1 [24] and approaches the theoretical optimum. On the more challenging image retrieval task, it comprehensively outperforms strong LLMs with 96× more parameters.

2. Related Work

Information retrieval (IR) aims to efficiently retrieve relevant information from user queries, generally expressed in text form [8, 18, 35-37]. With the advent of LLMs, IR pipelines are composed of four components: Query Rewriter, Retriever, Reranker, and Reader [50]. The Query Rewriter refines the initial user query to better capture information needs. Retrievers project queries into highdimensional vector spaces and compute relevance via inner product scoring. Rerankers apply matching methods and domain-specific objectives to refine candidate sets to improve result quality. Optionally, Readers process with LLMs retrieved information as an answer in natural language. LLMs can support different stages of the pipeline to better capture user intents and behavior [1]. We propose a dialogue-based framework for the Query Rewriter that leverages LLMs to generate context-aware questions. These questions are conditioned on the history of previous user answers and guide users through a dialogue to iteratively refine and improve search results.

LLMs as Query Rewriters. Query retrieval often suffers from ambiguity or incompleteness in user queries. Two common strategies to address this are query expansion and conversational query rewriting. Query expansion, enriches the original query with semantically related terms [16, 43]. Conversational Query Rewriters addresses user query ambiguity through iterative, multi-turn dialogues between the IR system and user [45, 48]. Traditional methods struggle with diverse or long-tailed dialogue sessions [7], whereas LLMs' contextual understanding allows them to manage complex interactions and generate versatile queries in formats such as questions or answer-incorporated passages [28]. Moreover, RAG-based LLMs integrate explicit retrieval modules that allow the model to access external knowledge to support query generation [3, 10, 15, 27, 41]. Building on these advances, our approach employs a Query Rewriter LLM that formulates retrieval queries through dialogue by generating questions and incorporating binary (yes/no) answers from the user's conversation history.

Reinforcement Learning. Query rewriters operate between users and retrieval modules, but this intermediary role often leaves them without a dedicated loss function for optimization [50]. This limitation motivates the use of RL to better align query generation with downstream retrieval objectives. RL incorporates feedback from multiple sources. Ranking models provide preference signals through good-bad pairs [29, 34]. Offline scoring mechanisms, such as BEQUE [31], estimate query utility based on retrieved items. Retrieval-metric rewards, inspired by DeepSeek-R1 [9] and adopted in DeepRetrieval [17], enable optimization even without supervised data. Collectively, these strategies allow query rewriters to learn policies that more effectively improve retrieval.

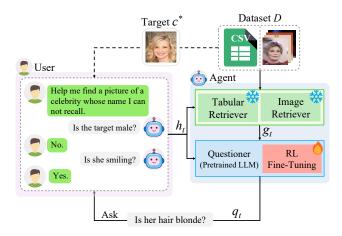


Figure 2. The overview of the pipeline. An LLM fine-tuned using reinforcement learning acts as a Questioner to ask questions, while the User answers questions as feedback.

Unlike prior work, we utilize an LLM-based Query Rewriter specifically designed as a *Questioner*, which we fine-tune using reinforcement learning from human feedback. Consequently, the Query Rewriter learns to generate the most relevant questions based on the conversation history and prompts the user with specific keyword attributes that are essential for successful retrieval. Overall, this framework enables the user to have only a vague idea of the query, while the LLM facilitates the interaction with the retriever to efficiently identify the target.

3. Method

SherlockLLM consists of an interactive agent designed to retrieve a target item c^* from a dataset D through a multiturn dialogue with a user. The agent's architecture is composed of two primary modules: a **Retriever** and a **Questioner** (Fig. 2). These modules operate sequentially to iteratively refine the search space and identify the target. At the beginning of an episode, a user has a target c^* without a full description. The user initiates the interaction with a free-form request. At dialogue turn t, the Retriever takes the current dialogue history h_t as input and queries D to obtain results g_t . Subsequently, the Questioner receives both the history h_t and the results g_t . It then generates the next yes/no question q_t conditioned on both g_t and h_t .

A key feature of SherlockLLM is its modularity. The learning mechanism of the Questioner is domain-agnostic, while the Retriever is domain-specific. This enables seamless adaptation to different retrieval modalities by replacing the retrieval and ranking components.

3.1. Questioner

A common strategy in dialogue-driven retrieval is to prepare a finite question pool and select a question via heuristics or

Algorithm 1 Keyword-Conditioned CLIP Retriever

Inputs: h_t (dialogue history), $I = \{I_1, \dots, I_n\}$ (images); hyperparameters μ (threshold), β (steepness), $d_0 \in (0,1)$ (base discount)

Output: Ranked image list by score S

14: $\mathcal{S} \leftarrow \mathcal{S}^+ \odot \mathcal{D}$

15: **return** SORT_{\downarrow}(\mathcal{S})

```
1: (K^+, K^-) \leftarrow \text{KEYWORDPARSER}(h_t)
       positive keywords K^+ and negative keywords K^-
      if |K^{+}| > 0 then
               t^+ \leftarrow \text{Join}(K^+, ",") \triangleright \text{Join positive keywords to a}
       query string
               S^+ \leftarrow CLIP(I, t^+)
  4:
  5: else
               \mathcal{S}^+ \leftarrow \mathbf{1}_N
  6:
  7: end if
  8: \mathcal{D} \leftarrow \mathbf{1}_N
                                                              > per-image discount vector
      \text{ for all } k^- \in K^- \text{ do }
              \begin{array}{l} \text{all } \kappa \in \Lambda \\ \mathcal{S}^- \leftarrow CLIP(I,k^-) \\ \psi \leftarrow 1 - (1-d_0) \cdot \sigma \big(\beta \left(\mathcal{S}^- - \mu\right)\big) \triangleright \sigma(x) = \frac{1}{1+e^{-x}} \\ & \triangleright \text{ elementwise product} \end{array}
10:
11:
13: end for
```

a learned network [12, 19, 44]. While effective in narrow settings, this design constrains the agent to a fixed hypothesis space: it cannot compose novel queries tailored to the unfolding dialogue, and transfers poorly across domains.

We therefore adopt a pretrained LLM as a dynamic Questioner. By leveraging broad world knowledge and generative capacity, the LLM can formulate coherent, strategic questions on the fly. We model the Questioner as a policy over questions conditioned on an augmented dialogue history that fuses user feedback with retrieval signals. We define the dialogue history up to turn t as a sequence of question—answer pairs:

$$h_t = \{(q_0, a_0), \dots, (q_{t-1}, a_{t-1})\}$$
 (1)

 q_t is a question produced by the agent, and a_t is the user's answer. To expose the Questioner to both the user's response and the Retriever's result, we define an *augmented* answer \tilde{a}_t :

$$\tilde{a}_t = a_t \oplus \rho\left(g_t\right) \tag{2}$$

⊳ final score per image

 \triangleright descending list by S

The function $\rho(\cdot)$ renders the retrieval output g_t into text (e.g., summarizing attribute distributions from tabular data or top-k retrieved images). The operator \oplus denotes string concatenation.

The Questioner is a policy π that generates the next question q_t given the augmented dialogue history \tilde{h}_t :

$$q_t \sim \pi \left(\cdot \mid \tilde{h}_t \right),$$
 (3)

The augmented history replaces each standard answer with its augmented counterpart:

$$\tilde{h}_t = \{(q_0, \, \tilde{a}_0), \, \dots, (q_{t-1}, \, \tilde{a}_{t-1})\}.$$
 (4)

3.2. Retriever

The retriever serves as the interface between the dialogue and the underlying dataset. Its role is to interpret h_t , update the agent's belief over candidates, and return the feedback to the questioner. The mechanism is similar to Retrieval-Augmented Generation (RAG). The concrete implementation is domain-specific to handle the unique characteristics of each data modality.

Tabular Retriever. For structured tabular datasets, the retriever operates deterministic logical filtering that contracts the hypothesis space at every turn. The agent's question q_t is first parsed to extract attribute—value pair(s). For instance, from the question "Is the target's hair blonde?" the parser yields $\{\text{hair_color:} \ \text{blonde}\}$. This can be implemented using either a rule-based system or a lightweight LLM. Given the attribute—value pair(s) and the user's answer a_t , the current candidate set \mathcal{C}_t is filtered. The retriever outputs a new, strictly smaller candidate set $|\mathcal{C}_{t+1}| < |\mathcal{C}_t|$. The distribution of \mathcal{C}_{t+1} is provided as feedback to the questioner. In addition, this pruned set is then used to calculate the EIG for the reward function.

Image Retriever. We formalize the image retriever as a *similarity-based ranker* that integrates positive and negative attributes to handle negation robustly. The retriever projects the user's queries and all candidate images $I = \{I_1, \ldots, I_n\}$ into a shared embedding space and returns a similarity score per image. Vision–language models (VLMs) such as CLIP [33] perform well on general retrieval tasks but often degrade on queries with negation (e.g., "a man without glasses"). To address this limitation, we propose a multi-stage retrieval process that explicitly models both positive and negative attributes.

The retriever first computes positive queries for every image. As detailed in Algorithm 1, our process begins by parsing the dialogue history h_t to obtain two sets of keywords: positives K^+ and negatives K^- . We concatenate positive keywords into a single text query t^+ , and compute the vector of positive similarities $\mathcal{S}^+ \in \mathbb{R}^n$, the *i*-th entry gives the CLIP similarity between t^+ and image I_i .

The retriever then converts each negative keyword into a vector of discounts \mathcal{D} that penalize images similar to that keyword. For every $k^- \in K^-$, we compute a similarity vector $\mathcal{S}^-(k^-) \in \mathbb{R}^n$ whose i-th entry is the CLIP similarity between negative keyword k^- and I_i . We transform this vector element-wise through a sigmoid gate $\sigma(\cdot)$ to obtain a discount factor $\psi(\mathcal{S}^-(k^-)) \in [d_0, 1]$:

$$\psi(x) = 1 - (1 - d_0) \cdot \sigma(\beta(x - \mu)) \tag{5}$$

where $d_0 \in (0,1)$ is a base discount, $\beta > 0$ defines the steepness, and $\mu \in \mathbb{R}$ is a similarity threshold. Intuitively, if an image has high similarity to a negative keyword, its corresponding discount factor $\psi(\cdot)$ approaches d_0 , thereby applying a bounded penalty. For images with low similarity, $\psi(\cdot)$ remains close to 1, imposing a negligible penalty.

We aggregate negative evidence by an element-wise product over all discount vectors. Let \odot denote the element-wise product. The aggregate discount vector \mathcal{D} is

$$\mathcal{D} = \bigodot_{k^- \in K^-} \psi\left(\mathcal{S}^-(k^-)\right), \ \mathcal{D} \in [d_0, 1]^n$$
 (6)

Finally, the initial positive score S^+ for each image is multiplied by its aggregate discount factor \mathcal{D} to compute the final score S. The retriever outputs a ranked list of all images, sorted in descending order based on S.

3.3. User

We use an LLM to simulate a user answering questions during training. Since our agent generates questions in an open-ended manner, a simple rule-based system for the simulator would be insufficient. The simulator workflow is as follows: First, it receives q_t and the complete attribute information of c^* , which is dynamically formatted into a prompt. For instance, the prompt will contain the target's data (e.g., {"gender": "female", "hair_color": "blonde", ...}) and the specific question posed by the agent. The prompt also includes a directive instructing the LLM to act as a truthful user, constraining its response to only: "Yes", "No", or "I can't answer", based strictly on the provided facts. This prompt is then sent to the simulator, which generates an answer a_t .

3.4. Reinforcement Learning

Our goal is to develop a dialogue agent that can efficiently identify a target item from a candidate set through a sequence of clarifying questions. We formulate this task as a finite Markov Decision Process (MDP), wherein a policy learns to generate questions that maximally reduce uncertainty about a hidden target.

Let $C=\{c_1,c_2,\ldots,c_n\}$ be a finite set of N candidate items. At the start of each dialogue episode, a single target item $c^*\in C$ is selected, which remains unknown to the agent. The agent's goal is to identify c^* by engaging in a multi-turn dialogue. The components of our RL framework are defined as follows:

State. At turn t, the state s_t is represented by the dialogue history:

$$s_t = \tilde{h}_t \tag{7}$$

where \tilde{h}_t is the history as defined in Eq. (4).

Action. The action is the generation of the next question q_t . This question is sampled from the policy $\pi_{\theta}(q_t|s_t)$. In our setting, the policy π_{θ} is implemented as an LLM.

Environment. The environment consists of a user simulator and a retriever module, which interact with the agent to generate the dialogue trajectory. An episode terminates when the agent correctly identifies the target c^* or after a maximum of T_{max} turns. The environment is also responsible for storing dialogue trajectories and providing a reward (Sec. 3.5) for each completed trajectory.

Learning Algorithm. To train the agent, we employ Group Relative Policy Optimization (GRPO) [39]. GRPO estimates the baseline using intra-group relative rewards, thereby avoiding the need for a separate value function model. This approach significantly reduces memory and computational resource consumption [39]. During each training iteration, we sample i targets. For each target c^* , we execute j stochastic rollouts, resulting in $i \times j$ trajectories, denoted as $\{\tau_{x,y}\}_{x=1...i,\ y=1...j}$. For each τ , the environment provides a trajectory-level reward r_{τ} . We then use these collected trajectories to estimate the policy gradient and update the policy parameters θ . This iterative process refines the agent's questioning strategy, making it more efficient and strategic over time.

3.5. Reward Functions

The reward function is a principal contributing factor affecting the performance of RL agents and is also challenging to design [4]. Because GRPO optimizes trajectory-level rewards, we assign each episode a single scalar reward that summarizes (i) a success bonus, (ii) the average step score, and (iii) a step-length penalty. Formally, let T be the number of turns in the episode $(1 \le T \le T_{\rm max})$ and let ϵ_t denote the step score at turn t. We define

$$\bar{\epsilon} = \frac{1}{T} \sum_{t=1}^{T} \epsilon_t, \tag{8}$$

and the trajectory reward:

$$\mathcal{R}(\tau) = \begin{cases} \kappa + \bar{\epsilon} - \text{Penalty}(T), & \text{success,} \\ -\kappa, & \text{otherwise,} \end{cases}$$
(9)

where $\kappa > 0$ is a success constant. The step-length penalty scales linearly with the episode length:

Penalty(T) =
$$\alpha \cdot \frac{T}{T_{\text{max}}}$$
, $\alpha \ge 0$, (10)

We define the step score ϵ_t differently for tabular and image settings.

Step Score for Tabular Data. The tabular setting is a structured, deterministic environment. Each candidate in the current pool is described by discrete attributes. Any attribute query deterministically partitions the current candidate set into two disjoint subsets, enabling an exact accounting of uncertainty reduction.

Table 1. Comparison to baseline for Guess Number task and Guess Who task. Oracle is the optimal baseline.

		Gues	s Numbe	r Gues	s Who
Model	Parameter	s SR↑	MT↓	SR↑	MT↓
Oracle	-	1.0	6.64	1.0	5.22
DeepSeek-V3.1	671B	1.0	8.02	1.0	6.18
	0.5B	0.0	16	0.0	16
O	1.5B	0.07	15.61	0.0	16
Qwen2.5	7B	0.68	12.43	0.46	13.26
	32B	0.92	9.39	0.56	11.44
Qwen2.5 + SFT	7B	0.88	10.38	0.48	11.66
$\overline{\text{Qwen2.5} + \text{GRPO}}$	7B	0.99	7.62	1.0	6.25

We measure uncertainty using Shannon entropy [38]. This method is a widely accepted direct reward for IR [13, 30, 47]. Let the unknown target be a random variable c supported on the current candidate set $\mathcal{C}_t = \{c_{1_t}, \ldots, c_{n_t}\}$ at turn t, with belief $p_t(x) = \Pr(c = x \mid s_t)$ given state s_t . The conditional entropy is

$$H(c \mid s_t) = -\sum_{x \in C_t} p_t(x) \log_2 p_t(x)$$
 (11)

Under a uniform belief, $H(c \mid s_t) = \log_2 n_t$. Let $a_t \in \{\text{yes}, \text{no}\}\$ the user's answer. The EIG of question q_t is:

$$EIG_t = H(c \mid s_t) - \mathbb{E}_{a_t \mid s_t} [H(c \mid s_t, a_t)]$$
 (12)

Since the answer deterministically partitions C_t into C_t^{yes} and C_t^{no} with sizes $n_t^{\text{yes}} = |C_t^{\text{yes}}|$ and $n_t^{\text{no}} = |C_t^{\text{no}}|$, the answer probabilities under a uniform prior are

$$p_t^{\text{yes}} = \frac{n_t^{\text{yes}}}{n_t}, \qquad p_t^{\text{no}} = 1 - p_t^{\text{yes}} = \frac{n_t^{\text{no}}}{n_t}$$
 (13)

The resulting step score is

$$\epsilon_t = \text{EIG}_t = \log_2 n_t - \left(p_t^{\text{yes}} \log_2 n_t^{\text{yes}} + p_t^{\text{no}} \log_2 n_t^{\text{no}} \right)$$
(14)

 ϵ_t is maximized for a balanced split ($p_t^{\rm yes} \approx 0.5$), achieving at most one bit of information per yes/no question.

Additionally, when a question fails the simulator-verified legality check (i.e., it is not a well-formed, answerable English interrogative), we bypass the EIG score and instead assign a soft penalty $\epsilon_t = \hat{\epsilon}$ with $-0.5 < \hat{\epsilon} < 0$.

Step Score for Image Data. In the image setting, the semantic space is high-dimensional and unstructured, making direct entropy estimation impractical. Instead, we track the agent's progress via the *rank of the ground-truth target* under a dialogue-conditioned retriever. A successful question should move the target closer to the top of the ranked list.

Table 2. Co	nparison to	baseline	for Celeb.	A image i	retrieval task.

Model	Parameters	Data Size	SR/R@5↑	MT↓	MedR↓	MR↓
DeepSeek-V3.1	671B	100 / 500	0.61 / 0.36	12.82 / 19.96	5.0 / 22.5	14.24 / 53.79
	1.5B	100 / 500	0.15 / 0.08	17.56 / 23.52	45.5 / 196.0	44.38 / 217.1
Qwen2.5	7B	100 / 500	0.36 / 0.23	15.33 / 21.52	17.0 / 65.0	25.75 / 115.30
	32B	100 / 500	0.46 / 0.24	13.90 / 20.89	8.5 / 49.5	20.80 / 89.42
Qwen2.5 + GRPO	7B	100 / 500	0.90 / 0.69	6.15 / 12.21	3.5 / 5.0	6.35 / 45.11

Given h_t at turn t, the retriever produces similarity scores for each candidate image and induces a total ordering Φ_t by sorting candidates in descending similarity. We focus on the rank of the true target c^* under this ordering:

$$\Phi_t \in \{1, 2, \dots, n\}, \qquad \phi_t = \operatorname{rank}_{\Phi_t}(c^*), \qquad (15)$$

We measure per-turn improvement by the change in *log-rank*:

$$\epsilon_t = \Delta_t = \log \phi_{t-1} - \log \phi_t \tag{16}$$

where $\epsilon_t > 0$, when the rank improves, $\epsilon_t = 0$, when it is unchanged, and $\epsilon_t < 0$, when it worsens. Similar to the tabular setting, we assign a soft penalty as a step score when an illegal question is generated.

4. Experiments

In this section, we conduct extensive experiments on two data modalities. We first introduce the datasets, and then present the experimental results and analysis.

4.1. Datasets

We evaluate SherlockLLM on two data modalities across three tasks. In all settings, an episode ends when the target is identified or the turn budget is exhausted.

Guess Number. This is a logical deduction task on a tabular dataset. In each game, a set of 100 consecutive integers is randomly chosen from a range of [0, 1000], and a single number is selected as the target.

Guess Who. Guess Who is a multi-attribute reasoning task on a tabular dataset. We use a dataset of 36 unique characters, each defined by 9 distinct attributes (more details in Sec. C).

CelebA Image Retrieval. This is a semantic retrieval task on an image dataset. CelebA [25] is a celebrity face attributes dataset containing 202,599 images annotated with 40 binary attributes. We remove blurred images and form two sets of 100 and 500 images to assess performance under small and larger search spaces. To create rich textual captions, we first use the binary attributes to create base captions and then rewrite them with GPT-40 [14] to improve fluency while preserving the encoded attributes. An episode is counted as successful when the target image is ranked within the top 5 (rank \leq 5) within T_{max} rounds.

4.2. Implementation Details

Model. We use Qwen2.5-7B-Instruct [42] as the question policy. To accelerate training and reduce computational overhead, we use a 4-bit quantized version of the model.

User Simulator. We use DeepSeek-V3.1 [24] to serve as the user simulator, responsible for answering questions.

Retriever. For our retriever, we set the hyperparameters in Eq. (5) to $\mu=0.15$, $\beta=20$, and $d_0=0.9$. These values were empirically determined to yield optimal performance. **Policy Learning.** We fine-tune Qwen2.5-7B-Instruct using LoRA [11] with a rank of 16 and an alpha of 32. The model is trained with the AdamW [26] optimizer with a learning rate of $2e^{-6}$. For the reward function, we set κ to 2 and the step penalty coefficient to 0.7. We generate 25 dialogue trajectories for each possible target within every task. The maximum number of turns T_{max} is task-dependent: for tabular tasks, T_{max} is 16. For image retrieval tasks, T_{max} is set to 20 for a dataset of 100 images and 25 for 500 images. All tasks are trained for 120 steps, except for the 500-image retrieval task, which is trained for 200 steps. For each task, we select the model that achieves the highest reward.

4.3. Baselines

State-of-the-Art LLM (Zero-Shot). We compare SherlockLLM against DeepSeek-V3.1 (671B parameters). This model is prompted in a zero-shot manner, using the same prompt and feedback format as our agent. For a fair comparison to other models, we do not use the "thinking mode". Foundation Models (Zero-Shot). We also compare SherlockLLM against its underlying models without any training in a zero-shot setting. We evaluate pretrained Qwen2.5-Instruct models of varying scales (0.5B, 1.5B, 7B and 32B). Supervised Fine-Tuning Method. We include a baseline developed using Supervised Fine-Tuning (SFT) on tabular tasks. It directly trains a model to imitate expert behavior by using a synthetic dialogue dataset. Training details are provided in Section D.1 and Section D.2.

Oracle. We define an Oracle baseline to represent the theoretical upper bound performance of our tabular tasks. The Oracle simulates a perfect decision-making process. At each round, it exhaustively searches the entire pool of single-attribute questions and selects the one that yields the

Task: find 143 from	186, 87,	184, 1851
---------------------	----------	-----------

No.	Question	Answer	EIG				
1	Is the target number less than or equal to 135?	No	1.0				
2	Is the target number less than or equal to 160?	Yes	1.0				
3	Is the target number less than or equal to 147?	Yes	1.0				
4	Is the target number less than or equal to 141?	No	1.0				
5	Is the target number less than or equal to 144?	Yes	1.0				
6	Is the target number 144?	No	0.92				
7	Is the target number 143?	Yes	1.0				
	(a) An example of the Guess Number task.						

Task: find C33 among 36 characters.

1 44.51	in this coo among or characters.		
No.	Question	Answer	EIG
1	Is the target candidate a male?	Yes	1.0
2	Is the target candidate's hair color blonde?	No	0.85
3	Is the target candidate's hairstyle bald?	No	1.0
4	Is the target candidate's hair color red?	No	0.92
5	Is the target candidate's hair color black?	No	1.0
6	Is the target candidate's hair color white and long hairstyle?	No	1.0
7	Is the target candidate C33?	Yes	0.0
	(b) An example of the Guess Who tasl	ζ.	

Figure 3. Examples from the Guess Number and Guess Who tasks. (a) illustrates the agent's strategy in Guess Number, while (b) shows its approach in Guess Who.

maximum EIG. This question is the most informative one, and ideally, it halves the remaining candidate set. This process continues until the target's probability becomes 1.

4.4. Metrics

For all tasks, we report the Success Rate (SR), which is the percentage of dialogues that successfully meet the task-specific completion criteria, and the Mean Turns (MT), which measures the mean number of turns for all dialogues. For the CelebA Image Retrieval task, we also report common retrieval metrics: Recall@K (R@K) for K=5, representing the percentage of trials where the target is ranked within the top 5. By this definition, R@5 is equivalent to SR. Finally, we report the Median Rank (MedR) and the Mean Rank (MR) of the target image across all trials.

4.5. Results and Analysis

Tabular Retrieval Tasks. Table 1 presents a performance comparison of SherlockLLM (Qwen2.5-7B + GRPO) against baseline models on tabular tasks. On Guess Number, the Oracle achieves a perfect SR in an optimal number of turns. SherlockLLM, while maintaining an almost perfect SR, yields an MT of 7.6. Compared to DeepSeek, SherlockLLM reduces MT by 5.0%. On the Guess Who task, both SherlockLLM and DeepSeek attain a 100% SR. Our policy requires an average of 6.25 turns per dialogue, a result that is on par with DeepSeek and one step

Target	No.	Question	Answer	Rank	EIG	EIG Rank
	1	Is the target a man?	No	53	0.98	1
36	2	Does the target have bangs?	No	10	0.88	2
	3	Does the target have earrings?	Yes	3	0.92	1

(a) An example of progressive refinement in the CelebA image retrieval task, where each question improves the target's rank.

Target	No.	Question	Answer	Rank	EIG	EIG Rank
	1	Is the target a man?	Yes	25	0.98	1
1000	2	Does the target have a goatee?	No	27	0.97	2
	3	Does the target have black hair?	No	32	0.54	4
	4	Does the target have brown hair?	Yes	7	0.76	3
	5	Does the target wear eyeglasses?	Yes	3	0.72	2

(b) A more challenging retrieval task. The agent demonstrates resilience by recovering from an initial rank degradation caused by noisy similarity.

Figure 4. Examples from the image retrieval task. For each turn, we show the target's rank, EIG of the chosen question, and its EIG Rank among all possible questions for the top-10 candidates.

more than the Oracle. The Qwen2.5 models without fine-tuning underperform. The smallest variants (0.5B and 1.5B) fail to complete the tasks, indicating insufficient reasoning and instruction-following ability at this scale. Even the 32B variant stops at SR = 0.92/0.56 with MT = 9.39/11.44 on the two tasks. We compare the Qwen2.5-7B backbone against its SFT and GRPO counterparts. On Guess Number, SFT method increases SR by 29% and reduces MT by 16.5%. GRPO-tuned agent improves SR by 45.6% and reduces MT by 38.7%. On Guess Who, SFT provides minimal improvement to the baseline model's SR and MT, showing improvements of 5.0% and 12.1%, respectively. For the GRPO method, improvements are even more striking: SR more than doubles from 0.46 to 1.0, with MT dropping by 52.9%.

CelebA Image Retrieval. SherlockLLM achieves state-of-the-art performance on the CelebA image retrieval task, outperforming all baseline models (Tab. 2). Compared to DeepSeek-V3.1, SherlockLLM shows a substantial advantage in dialogue efficiency, reducing MT by approximately 52% and 38.8% on the two test sets. In addition, its SR is significantly higher, showing improvements of 47.5% and 91.7% over DeepSeek-V3.1. Compared to the Qwen2.5-7B base model, the GRPO training yields remarkable gains: SR increases by 150% and 200% on the 100- and 500-image sets, respectively. The training also improves ranking performance, with MedR decreasing by 79.4% and 92.3%, and MR decreasing by 75.3% and 60.9%. Concurrently, dialogue efficiency is enhanced, as evidenced by a reduction in MT of 59.9% and 43.3% on the respective sets.

Qualitative examples. In Guess Number and Guess Who (Fig. 3), SherlockLLM learns near-optimal questioning strategies. For instance, in Guess Number, its approach is analogous to a binary search, consistently halving the candidate space to achieve a maximal EIG of 1.0. Similarly, in Guess Who, SherlockLLM learns to prioritize

Table 3. Ablation study on different reward functions.

Reward Function	SR↑	MT↓
EIG Only	0.99	7.55
Step Penalty Only	0.99	6.19
EIG + Step Penalty	1.0	6.25

broad, discriminative questions about high-level attributes such as gender. These cases demonstrate that in well-defined problem spaces, our framework effectively guides the agent toward mathematically efficient policies. In the image retrieval, Figure 4a shows a progressive refinement, where each subsequent question steadily improves the target's rank. Fig. 4b reveals SherlockLLM's ability to handle non-monotonic progression. In this challenging example, a negative answer temporarily worsens the target's rank from 25 to 32. This is a likely consequence of the complexities within the visual-semantic embedding space. However, SherlockLLM recovers from this setback, ultimately improving the target's rank to 3. This self-correction capability is a key advantage of our interactive approach over traditional one-shot query retrieval systems.

Ablation Study on Reward Components. To ascertain the individual contribution of each component within our reward function (Sec. 3.5), we conduct an ablation study for the Guess Who task. We use three experimental conditions: 1) an agent rewarded solely based on EIG, 2) an agent guided by a terminal reward and a step penalty, and 3) an agent trained with the complete reward function. The success-contingent constant term is retained across all configurations. The results are presented in Tab. 3. Sherlock-LLM trained with only the EIG reward learns to ask highly informative questions. However, this leads to a deliberate but slow strategy. SherlockLLM guided exclusively by a step penalty becomes highly efficient (6.19 MT), but optimized purely for speed, may not always be the most robust approach. The combination of both components in the reward yields a more robust policy. This agent achieves a perfect success rate while maintaining a highly efficient average of 6.25 turns. The per-step guidance from the EIG reward teaches the agent what constitutes an effective query, while the global step penalty incentivizes it to reach the goal expeditiously. This combination compels the agent to learn a strategy that is both effective and temporally efficient.

Impact of Retrieval Feedback. Table 4 isolates the effect of retrieval feedback. For Guess Who, injecting attribute-distribution feedback shortens the dialogue while maintaining ceiling success. This indicates that the agent already solves the task reliably, and feedback mainly helps it ask more informative follow-ups. For the image retrieval, the choice of top-K feedback is decisive. Moving from top-5 to top-10 retrieved images boosts SR by 36.4%, and cuts MR

Table 4. **Impact of retrieval feedback.** For the Guess Who, we compare a model with no feedback against one that receives the current attribute distribution. For the image retrieval, we compare the models using the top-5 or top-10 retrieved images.

Task	Feedback	SR↑	MT↓	MedR↓	MR↓
Guess Who	No Feedback	1.0	6.98	-	-
	Feedback	1.0	6.25	-	-
Image Retrieval	Top-5	0.66	9.92	$-\frac{1}{4.0}$	16.7
illiage Keuleval	Top-10	0.90	6.15	3.5	6.35

Table 5. Comparison to different image retrieval backbones.

Models	Data Size	SR/R@5↑	$\mathbf{MT}\!\!\downarrow$	$\mathbf{MedR}{\downarrow}$	MR↓
CLIP	100 / 500	0.90 / 0.69	6.15 / 12.21	3.5 / 5.0	6.35 / 45.11
BLIP	100 / 500	1.0 / 0.27	5.22 / 20.44	2.0 / 64.0	2.36 / 112.11

by 62.0%. Shrinking K from 10 to 5 amplifies the noise and bias in the model's estimation of image feature distribution based on top-K captions, leading to inaccurate expected information gain. Increasing K can stabilize this distribution estimation, enabling the strategy to propose clearer and more valuable queries.

Impact of Retrieval Backbone. We investigate the impact of the retrieval backbone by comparing a training-free CLIP retriever and a fine-tuned BLIP [22] counterpart. BLIP is first fine-tuned on a synthetic dialogue dataset (Sec. D.3). The experiments are conducted on 100-image and 500-image set while holding the Questioner module and RL framework constant. As shown in Tab. 5, the BLIP backbone achieves a higher ranking quality for the 100-image set. However, this trend reversed in the 500-image setting. While CLIP's success rate remains stable at 0.69, BLIP's performance degrades to 0.27. A larger dataset demands longer dialogues and more compositional questions. The end-to-end BLIP does not generalize well, whereas the keyword-based CLIP demonstrates more robust scalability.

5. Conclusions

We propose a dialogue-driven information retrieval framework trained with reinforcement learning. Our method trains an agent to ask a sequence of optimally informative questions to efficiently narrow the search space. This approach aims to improve the retrieval accuracy and reduce the interaction turns. To validate our framework, we design three benchmark tasks across two data modalities: two structured data retrieval games and an image retrieval task. Experimental results demonstrate that our method significantly outperforms strong baselines across all tasks.

References

- [1] Qingyao Ai, Ting Bai, Zhao Cao, Yi Chang, Jiawei Chen, Zhumin Chen, Zhiyong Cheng, Shoubin Dong, Zhicheng Dou, Fuli Feng, et al. Information retrieval meets large language models: a strategic report from chinese ir community. *AI open*, 2023. 2
- [2] Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. Asking clarifying questions in opendomain information-seeking conversations. In SIGIR, 2019.
- [3] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR, 2022. 2
- [4] Yuji Cao, Huan Zhao, Yuheng Cheng, Ting Shu, Yue Chen, Guolong Liu, Gaoqi Liang, Junhua Zhao, Jinyue Yan, and Yun Li. Survey on large language model-enhanced reinforcement learning: Concept, taxonomy, and methods. In IEEE Transactions on Neural Networks and Learning Systems, 2025. 5
- [5] Peiyuan Chen, Zecheng Zhang, Yiping Dong, Li Zhou, and Han Wang. Enhancing visual question answering through ranking-based hybrid training and multimodal fusion. *arXiv* preprint arXiv:2408.07303, 2024. 1
- [6] Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In SIGIR, 2009.
- [7] Zhuyun Dai, Arun Tejasvi Chaganty, Vincent Y Zhao, Aida Amini, Qazi Mamunur Rashid, Mike Green, and Kelvin Guu. Dialog inpainting: Turning documents into dialogs. In International conference on machine learning, pages 4558– 4586. PMLR, 2022. 2
- [8] Ritendra Datta, Dhiraj Joshi, Jia Li, and James Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. In ACM Comput. Surv., 2008. 2
- [9] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv, 2025. 2
- [10] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pretraining. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020. 2
- [11] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 6
- [12] Huang Hu, Xianchao Wu, Bingfeng Luo, Chongyang Tao, Can Xu, Wei Wu, and Zhan Chen. Playing 20 question game with policy-based reinforcement learning. In *EMNLP*, 2018.
- [13] Zhiyuan Hu, Chumin Liu, Xidong Feng, Yilun Zhao, See-Kiong Ng, Anh Tuan Luu, Junxian He, Pang Wei Koh, and Bryan Hooi. Uncertainty of Thoughts: Uncertainty-

- Aware Planning Enhances Information Seeking in LLMs. In *NeurIPS*, 2024. 2, 5
- [14] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. In arXiv preprint arXiv:2410.21276, 2024. 6
- [15] Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43, 2023. 2
- [16] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. Query expansion by prompting large language models. arXiv preprint arXiv:2305.03653, 2023. 2
- [17] Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. CoRR, 2025. 2
- [18] Vladimir Karpukhin, Barlas Oguz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. In EMNLP, 2020. 2
- [19] Sedrick Keh, Justin T Chiu, and Daniel Fried. Asking more informative questions for grounded retrieval. In *Findings* of the Association for Computational Linguistics: NAACL 2024, 2023. 3
- [20] Saehyung Lee, Sangwon Yu, Junsung Park, Jihun Yi, and Sungroh Yoon. Interactive text-to-image retrieval with large language models: A plug-and-play approach. In 62nd Annual Meeting of the Association for Computational Linguistics, 2024. 2
- [21] Matan Levy, Rami Ben-Ari, Nir Darshan, and Dani Lischinski. Chatting makes perfect: Chat-based image retrieval. In *NeurIPS*, 2023. 2
- [22] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *PMLR*, 2022. 8
- [23] Stella Li, Vidhisha Balachandran, Shangbin Feng, Jonathan Ilgen, Emma Pierson, Pang Wei W Koh, and Yulia Tsvetkov. Mediq: Question-asking Ilms and a benchmark for reliable interactive clinical reasoning. In *NeurIPS*, 2024. 2
- [24] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. arXiv preprint arXiv:2412.19437, 2024. 2, 6
- [25] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015.
- [26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *arXiv preprint arXiv:1711.05101*, 2017. 6
- [27] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting in retrieval-augmented large language models. In EMNLP, 2023. 2

- [28] Kelong Mao, Zhicheng Dou, Fengran Mo, Jiewen Hou, Haonan Chen, and Hongjin Qian. Large language models know your contextual search intent: A prompting framework for conversational search. In EMNLP, 2023. 2
- [29] Shengyu Mao, Yong Jiang, Boli Chen, Xiao Li, Peng Wang, Xinyu Wang, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Rafe: Ranking feedback improves query rewriting for rag. In EMNLP, 2024. 2
- [30] Davide Mazzaccara, Alberto Testoni, and Raffaella Bernardi. Learning to ask informative questions: Enhancing LLMs with preference optimization and expected information gain. In Findings of the Association for Computational Linguistics: EMNLP 2024, 2024. 5
- [31] Wenjun Peng, Guiyang Li, Yue Jiang, Zilong Wang, Dan Ou, Xiaoyi Zeng, Derong Xu, Tong Xu, and Enhong Chen. Large language model based long-tail query rewriting in taobao search. In *ACM*, 2024. 2
- [32] Zackary Rackauckas. Rag-fusion: a new take on retrievalaugmented generation. arXiv preprint arXiv:2402.03367, 2024. 1
- [33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 4
- [34] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*, 2023. 2
- [35] Stephen Robertson, Hugo Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 2009. 2
- [36] Gerard Salton and Michael J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., USA, 1986.
- [37] Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the* ACM, 1975. 2
- [38] Claude E Shannon. A mathematical theory of communication. In *The Bell system technical journal*, 1948. 5
- [39] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. In arXiv preprint arXiv:2402.03300, 2024. 5
- [40] Clemencia Siro, Yifei Yuan, Mohammad Aliannejadi, and Maarten de Rijke. Agent-cq: Automatic generation and evaluation of clarifying questions for conversational search with llms. In arXiv preprint arXiv:2410.19692, 2024. 2
- [41] Krishna Srinivasan, Karthik Raman, Anupam Samanta, Lingrui Liao, Luca Bertelli, and Michael Bendersky. Quill: Query intent with large language models using retrieval augmentation and multi-stage distillation. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track, 2022. 2
- [42] Qwen Team. Qwen2.5 technical report. In arXiv preprint arXiv:2412.15115, 2024. 6
- [43] Liang Wang, Nan Yang, and Furu Wei. Query2doc: Query expansion with large language models. In *EMNLP*, 2023. 2

- [44] Julia White, Gabriel Poesia, Robert Hawkins, Dorsa Sadigh, and Noah Goodman. Open-domain clarification question generation without question examples. In EMNLP, 2021. 3
- [45] Raymond Wilson, Chase Carter, and Cole Graham. Contextualizing search queries In-context learning for conversational rewriting with LLMs. In arXiv preprint arXiv:2502.15009, 2025. 1, 2
- [46] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. In *Transactions of the Association for Computational Linguistics*, 2014. 9
- [47] Lili Yu, Howard Chen, Sida I. Wang, Tao Lei, and Yoav Artzi. Interactive classification by asking informative questions. In 58th Annual Meeting of the Association for Computational Linguistics, 2020. 5
- [48] Yifei Yuan, Zahra Abbasiantaeb, Mohammad Aliannejadi, and Yang Deng. Query understanding in Ilm-based conversational information seeking. In Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 4098–4101, 2025. 1,
- [49] Hamed Zamani, Bhaskar Mitra, Everest Chen, Gord Lueck, Fernando Diaz, Paul N Bennett, Nick Craswell, and Susan T Dumais. Analyzing and learning from user interactions for search clarification. In SIGIR, 2020. 1
- [50] Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zheng Liu, Zhicheng Dou, and Ji-Rong Wen. Large language models for information retrieval: A survey. ACM Trans. Inf. Syst., 2025. 2

Sherlock Your Queries: Learning to Ask the Right Questions for Dialogue-Based Retrieval

Supplementary Material

A. Analysis of Alternative Methods

This section justifies the core design choices of the Sher-lockLLM for image retrieval. First, we justify our use of a rank-based reward function, which we compare to an alternative similarity-based formulation. Second, we evaluate our ranking algorithm. This evaluation involves a comparison with other common fusion methods.

A.1. Image Retrieval Reward Signal.

To optimize the training of our agent for the interactive image retrieval task, we investigated the efficacy of two different reward signal formulations. Intuitively, if the similarity score between the target and the query is higher, then it will rank higher. The objective of this experiment was to determine whether a direct optimization of the target's rank is more effective than using an implicit signal based on textimage similarity. All comparative experiments were trained on the 100-image dataset. Except for the definition of the reward function, all other conditions and training procedures remained consistent.

As Table 6 shows, both reward signals achieve an identical SR of 0.90. This indicates that both formulations are feasible. However, we observe clear differences in efficiency and ranking quality. The agent trained with the Rank-based reward demonstrates superior efficiency. It requires an MT of 6.15, a 6.1% reduction in conversational length compared to the 6.55 turns required by the agent trained on the Similarity score. Furthermore, the Rank-based reward achieves a MedR of 3.5, representing a 12.5% improvement for the Similarity-based reward.

Interestingly, the Similarity-based reward achieves a slightly better MR of 5.9. The Rank-based reward's MR is approximately 7.6% worse by comparison. We hypothesize that while the similarity-based agent is less efficient on average, it generates more exceptionally descriptive questions that result in a perfect R@1 retrieval. These highly successful instances would significantly pull down the mean rank. Conversely, the rank-based agent is more consistent, reliably placing the target in the top few positions as suggested by its superior median rank, but with fewer perfect R@1 retrievals. This is substantiated by an analysis of the R@1 metric. The agent trained on Similarity achieves an R@1 of 0.25, whereas the agent trained on Rank only reaches 0.1.

Despite the superior mean rank achieved with the similarity signal, we select the Rank-based reward as the main method in this work. The significant gains in dialogue efficiency and more reliable typical performance (12.5% better

Table 6. Comparison of retrieval performance on CelebA image retrieval task using Rank vs. Similarity as the reward signal.

Reward Signal	SR/R@5↑	MT↓	MedR↓	MR↓	R@1↑
Rank Similarity	0.90 0.90	6.15 6.55	3.5 4.0	6.35 5.9	0.1 0.25

median rank) are of paramount importance for an interactive retrieval system.

A.2. Ranking Algorithm Comparison

The ranking algorithm is a critical component of our agent for the image retrieval task. It translates the dialogue history into a ranked list of candidate images at each turn. We justify our choice of algorithm by comparing our proposed method against two plausible and computationally efficient alternatives.

Our primary ranking algorithm, detailed in Algorithm 1, operates on a cumulative basis. This method is designed for high precision because it leverages all information gathered up to the current turn. The main drawback of this approach is its computational cost. Recalculating similarities for the entire keyword history at every turn can introduce significant latency. We therefore design two alternative algorithms that improve efficiency by avoiding these historical recalculations.

Reciprocal Rank Fusion (RRF). RRF [6] is an algorithm that combines multiple ranked lists into a single, more optimal result. This method has seen wide application in modern retrieval systems [5, 32]. The algorithm is entirely training-free. It operates only on the rank positions of items and does not require their underlying relevance scores. The RRF score for a candidate is the sum of the reciprocal of its rank across all lists. A small constant k=60 is typically added to the denominator. The formula is given by the following equation.

$$Score_{RRF}(c) = \sum_{i=1}^{N} \frac{1}{k + rank_i(c)}$$
 (17)

where $rank_i(c)$ is the rank of candidate c in the i-th list from a total of N lists.

In our implementation of this alternative, we use RRF to fuse information sequentially across dialogue turns. At each turn t, we apply Algorithm 1 using only the most recent keywords. This process generates a turn-specific ranked list,

denoted as $\operatorname{Rank}_t = \{\operatorname{rank}_1, \operatorname{rank}_2, \dots, \operatorname{rank}_t\}$. The consolidated ranking for the current turn is then produced by applying Equation (17) to the complete set of lists generated up to that point. This approach allows for turn-by-turn updates without the computational overhead of re-evaluating the entire dialogue history.

Sequential Score Multiplication (SSM). SSM also operates sequentially but fuses information at the score level. This method functions as a soft logical AND gate. A candidate image must satisfy the constraints from every turn to remain a viable option.

At each turn t, a turn-specific score is calculated for each candidate image based on its similarity to the current turn's positive keywords ($Sim_{positive}$) and negative keywords ($Sim_{negative}$), with both values scaled to a [0,1] range. The turn score is then defined by:

$$Score_t = Sim_{positive} \times (1 - Sim_{negative})$$
 (18)

The final relevance score for an image at turn t is the cumulative product of all turn scores up to that point, where $S_{final,t} = S_{final,t-1} \times Score_t$. The initial score is set to $S_{final,0} = 1$. This cumulative multiplication ensures that only images that consistently receive high scores across all turns will maintain a high final ranking.

Results. We conduct the experiment on the 100-image dataset. The dialogues are generated for each target image using the methodology described in Section D.3.

As shown in Table 7, the results demonstrate the clear superiority of our proposed algorithm across all key metrics. Our SherlockLLM algorithm achieves an R@5 of 0.42, a R@10 of 0.57, and a MedR of 7.5. In contrast, the performance of the two methods is significantly worse. The RRF algorithm's R@5 is only 0.06, while the SSM algorithm fails to rank any target image in the top position. Their Median and Mean Ranks both exceed 50. This result indicates the target image did not even enter a meaningful ranking range.

The failure of the RRF method stems from its incorrect assumption that information from each turn constitutes parallel and independent evidence. In our interactive scenario, the dialogue imposes sequential, conditional constraints. For example, a query for "red hair" after a positive response to "is it a man?" implies a search for a "man with red hair." The RRF fusion mechanism cannot model this implicit AND condition. Instead, it simply combines the ranked list for "man" with the ranked list for "red hair," failing to properly constrain the search space as the dialogue progresses.

The SSM method fails because it operates on the flawed assumption that similarity scores from each turn are equally

Table 7. Comparison to different image retrieval ranking algorithms.

Algorithm	R@1↑	R@5↑	R@10↑	MedR↓	MR↓
SherlockLLM	0.15	0.42	0.57	7.5	15.33
RRF	0.02	0.06	0.1	50.5	50.41
SSM	0.0	0.01	0.05	53.5	53.73

important and directly comparable. However, query terms vary in their semantics and difficulty, which leads to vastly different score distributions from the retrieval model. Multiplying these scores, which are derived from different scales, distorts the true relevance of candidates and produces unreliable rankings.

B. Additional Case Studies

B.1. Guess Number

Figure 5 shows examples of three models on the Guess Number task. The goal is to find the target number 143 within an integer set {86, ..., 185}. The DeepSeek and Qwen2.5-32B execute a near-optimal binary search. In contrast, Qwen2.5-1.5B performs an inefficient linear scan by sequentially checking fixed-size intervals. While it correctly identifies the target's range in turn 9, the model fails to narrow the search further and exceeds the maximum turn limit.

The Qwen2.5-7B attempts a binary search but demonstrates a flawed strategy. The model commits severe logical errors and repetitive questions. This indicates that 7B can imitate the form of an optimal policy, but lacks the robust reasoning for the task.

SFT agent learns to mimic actions from the training data, but it fails to optimize for turn efficiency. In the initial phase (turns 1-6), the agent's performance is optimal. It consistently poses a series of questions with high or perfect EIG scores. However, the agent's strategy falters in the endgame phase. At turn 7, instead of making a decisive guess, the agent asks other questions.

B.2. Guess Who

Figure 6 presents a comparison of the dialogue strategies that the three models employed on the Guess Who task. The DeepSeek demonstrates a logical and efficient strategy. It queries attributes step by step and consistently selects questions with a high EIG. In contrast, the Qwen2.5-1.5B exhibits a flawed policy. It enters a repetitive loop, repeatedly asking a question with zero EIG. This behavior indicates a deficiency in multi-turn conversation capabilities and ultimately results in task failure.

The Qwen2.5-7B model shows an improvement over its 1.5B counterpart. It initially asks general attributes questions. However, while target C33 was the only remaining

	(a) Qwen2.5-7B+GRPO				(b) DeepSeek-V3.1		
No.	Question	Answer	EIG	No.	Question	Answer	EIG
1	Is the target number less than or equal to 135?	No	1.0	1	Is the target number greater than 135?	Yes	1.0
2	Is the target number less than or equal to 160?	Yes	1.0	2	Is the target number greater than 160?	No	1.0
3	Is the target number less than or equal to 147?	Yes	1.0	3	Is the target number greater than 147?	No	0.99
4	Is the target number less than or equal to 141?	No	1.0	4	Is the target number greater than 141?	Yes	1.0
5	Is the target number less than or equal to 144?	Yes	1.0	5	Is the target number greater than 144?	No	1.0
6	Is the target number 144?	No	0.92	6	Is the target number greater than 142?	Yes	0.91
7	Is the target number 143?	Yes	1.0	7	Is the target number 143?	Yes	1.0
	(c) Qwen2.5-1.5B				(d) Qwen2.5-7B		
No.	Question	Answer	EIG	No.	Question	Answer	EIG
1	Is the target number less than 90?	No	0.24	1	Is the target number less than or equal to 135?	No	1.0
2	Is the target number between 86 and 93 (inclusive)?	No	0.0	2	Is the target number less than or equal to 180?	Yes	0.46
3	Is the target number between 90 and 97 (inclusive)?	No	0.41	3	Is the target number less than or equal to 157?	Yes	0.99
4	Is the target number between 98 and 105 (inclusive)?	No	0.43	4	Is the target number less than or equal to 121?	No	0.0
5	Is the target number between 106 and 113 (inclusive)?	No	0.46	5	Is the target number less than or equal to 144?	Yes	0.97
6	Is the target number between 114 and 121 (inclusive)?	No	0.50	6	Is the target number less than or equal to 139?	No	0.99
7	Is the target number between 122 and 129 (inclusive)?	No	0.54	7	Is the target number less than or equal to 148?	Yes	0.0
8	Is the target number between 130 and 137 (inclusive)?	No	0.59	8	Is the target number less than or equal to 146?	Yes	0.0
9	Is the target number between 138 and 145 (inclusive)?	Yes	0.65	9	Is the target number less than or equal to 144?	Yes	0.0
10	Congratulations! You"ve guessed the correct number: 145.	I can't answer this question	0.0	10	Is the target number less than or equal to 142?	No	0.97
11	Is the target number between 146 and 153 (inclusive)?	No	0.0	11	Is the target number less than or equal to 145?	Yes	0.0
12	Is the target number between 154 and 161 (inclusive)?	No	0.0	12	Is the target number less than or equal to 144?	Yes	0.0
13	Is the target number between 162 and 169 (inclusive)?	No	0.0	13	Is the target number 144?	No	1.0
14	Is the target number between 170 and 177 (inclusive)?	No	0.0	14	Is the target number less than or equal to 143?	Yes	0.0
15	Is the target number between 178 and 185 (inclusive)?	No	0.0	15	Is the target number 143?	Yes	0.0
16	Is the target number exactly 185?	No	0.0				
	(e) Qwen2.5-32B				(f) Qwen2.5-7B + SFT		
No.	Question	Answer	EIG	No.	Question	Answer	EIG
1	Is the number even?	No	1.0	1	Is this number less than or equal to 135?	No	1.0
2	Is the number less than 135?	No	0.99	2	Is this number odd?	Yes	1.0
3	Is the number less than 165?	Yes	0.98	3	Is this number greater than 160?	Yes	0.99
4	Is the number less than 150?	Yes	0.99	4	Is this number greater than 147?	No	1.0
5	Is the number less than 142?	No	1.0	5	Is this number greater than or equal to 142?	Yes	1.0
6	Is the number less than 146?	Yes	1.0	6	Is this number greater than 144?	No	0.92
7	Is the number 143?	Yes	1.0	7	Is this number greater than or equal to 143?	Yes	0.0
				8	Is this number greater than 143?	No	0.0
				9	This number: 143.	Yes	0.0

Figure 5. Examples of the Guess Number performed by baseline models and SherlockLLM.

candidate after turn 6, the model proceeded with several rounds of unnecessary guessing before making the correct identification.

Qwen2.5-32B is better than Qwen2.5-7B. It asks high-EIG questions about character attributes. However, its strategy is demonstrably suboptimal and contains clear logical inconsistencies. It fails to perform the necessary logical deduction on the constrained candidate pool.

Similar to the behavior of SFT in Guess Number task, SFT agent can mimic questioning, but its policy is weak when the optimal action shifts from information gathering to logical inference.

(a) Qwen2.5-7B+GRPO					(b) DeepSeek-V3.1					
No.	Question	Answer	EIG	No.	Question	Answer	EIG			
1	Is the target candidate a male?	Yes	1.0	1	Is the target male?	Yes	1.0			
2	Is the target candidate's hair color blonde?	No	0.85	2	Does the target have a beard?	Yes	0.91			
3	Is the target candidate's hairstyle bald?	No	1.0	3	Does the target have a bald hairstyle?	No	1.0			
4	Is the target candidate's hair color red?	No	0.92	4	Is the target wearing glasses?	Yes	0.91			
5	Is the target candidate's hair color black?	No	1.0	5	Is the target's hair color white?	No	1.0			
6	Is the target candidate's hair color white and long hairstyle?	No	1.0	6	I guess the target is C33.	Yes	0.0			
7	Is the target candidate C33?	Yes	0.0							
	(c) Qwen2.5-1.5B				(d) Qwen2.5-7B					
No.	Question	Answer	EIG	No.	Question	Answer	EIG			
1	Is this person's occupation a teacher?	No	0.76	1	Is the target candidate a male?	Yes	1.0			
2	Does the target candidate enjoy reading?	No	0.59	2	Is the target candidate's hair color blonde?	No	0.85			
3	Is the target candidate wearing earrings?	Yes	1.0	3	Is the target candidate's hair color white?	No	0.89			
4	Is the target candidate a chef?	No	0.91	4	Is the target candidate's hair color red?	No	0.91			
5	Is the target candidate wearing earrings?	Yes	0.0	5	Is the target candidate's hairstyle bald?	No	1.0			
6	Is the target candidate wearing earrings?	Yes	0.0	6	Is the target candidate's hair color black?	No	0.91			
7	Is the target candidate wearing earrings?	Yes	0.0	7	Is the target candidate's hair color brown?	Yes	0.0			
8	Is the target candidate wearing earrings?	Yes	0.0	8	Is the target candidate's occupation a student?	Yes	0.0			
9	Is the target candidate wearing earrings?	Yes	0.0	9	Is the target candidate's hairstyle curly?	Yes	0.0			
10	Is the target candidate wearing earrings?	Yes	0.0	10	Is the target candidate C03?	No	0.0			
11	Is the target candidate wearing earrings?	Yes	0.0	11	Is the target candidate C09?	No	0.0			
12	Is the target candidate wearing earrings?	Yes	0.0	12	Is the target candidate C14?	No	0.0			
13	Is the target candidate wearing earrings?	Yes	0.0	13	Is the target candidate C23?	No	0.0			
14	Is the target candidate wearing earrings?	Yes	0.0	14	Is the target candidate C33?	Yes	0.0			
15	Is the target candidate wearing earrings?	Yes	0.0		8					
16	Is the target candidate wearing earrings?	No	0.0							
	is the target candidate wearing carrings.	1.0	0.0							
	(e) Qwen2.5-32B				(f) Qwen2.5-7B + SFT					
	Question	Answer	EIG	No.	Question	Answer	EIG			
1	Is the target candidate a male?	Yes	1.0	1	Is the person in question a man?	Yes	1.0			
2	Does the target candidate have a beard?	Yes	0.92	2	Does the character have a beard?	Yes	0.92			
3	Does the target candidate have brown eyes?	No	0.92	3	Is it true that the individual's hair is either blonde or white?	No	0.98			
4	Does the target candidate have amber eyes?	Yes	0.95	4	Can you confirm if the person's eyes are brown?	No	0.97			
5	Does the target candidate wear glasses?	Yes	0.0	5	the target is C36.	No	0.92			
6	Does the target candidate have brown hair?	Yes	0.92	6	the target is C25.	No	0.0			
7	Does the target candidate have a curly hairstyle?	Yes	1.0	7	the target is C23.	No	0.0			
8	C33	Yes	0.0	8	the target is C35.	No	0.0			
_				9	the target is C33.	Yes	1.0			
				_			1.0			

Figure 6. Examples of the Guess Who task performed by baseline models and SherlockLLM. Each example presents the full dialogue history for identifying the target C33.

B.3. CelebA Image Retrieval

Figure Fig. 7 illustrates the model behaviors on the CelebA image retrieval task. DeepSeek uses a diverse and exploratory strategy, similar to its performance on the Guess Who task. It successfully completes the task by querying a wide range of attributes. In contrast, Qwen2.5-1.5B exhibits a failure by generating severe hallucinations after several initial turns, repeatedly asking for attributes that do not exist in the dataset, such as "tattoos" and "scars". Qwen2.5-7B demonstrates a distinct failure mode: it continuously asks the same questions. This repetitive behavior halts the retrieval process, ultimately leading to task failure.

C. Guess Who Dataset

We design a set of 36 characters (C01–C36), with each character annotated by 9 discrete attributes shown in Tab. 8. The complete character set is detailed in Tab. 9. To quantify the distributional balance of each attribute, we use normalized entropy (NE), where a value of 1.0 indicates a perfectly uniform distribution. As shown in Fig. 8, all attributes achieve NE = 1.0 except has beard. The lower NE for the has beard attribute is a result of our modeling constraint that only male characters can have beards.

Target	No.	Question	Answer	Rank
	1	Is the person wearing eyeglasses?	No	10
	2	Is the person male?	No	51
10	3	Does the person have bangs?	Yes	61
	4	Does the person have heavy eye-makeup?	No	57
	5	Does the person have wavy hair?	No	49
	6	Does the person have an oval-shaped face?	Yes	40
	7	Does the person have a receding hairline?	No	42
	8	Does the person have a slightly open mouth?	Yes	38
	9	Does the person have a smile?	Yes	13
	10	Does the person have blond hair?	No	4
		(a) DeepSeek model.		

		() I		
Target	No.	Question	Answer	Rank
	1 2	Is the target face male? Does the target face wear glasses?	No No	58 51
	3	Does the target face have any piercings?	I can't answer this question	51
	4	Is the target face wearing jewelry?	Yes	32
	5	Does the target face have any tattoos?	I can't answer this question	32
	6	Does the target face have any scars?	I can't answer this question	32
	7	Does the target face have any birthmarks?	I can't answer this question	32
	8	Does the target face have any moles?	I can't answer this question	32
	9	Does the target face have any birthmarks or scars?	I can't answer this question	32
	10	Does the target face have any scars or birthmarks?	I can't answer this question	32
	11	Does the target face have any tattoos or scars?	I can't answer this question	32
	12	Does the target face have any scars or birthmarks?	I can't answer this question	32
	13	Does the target face have any birthmarks or scars?	I can't answer this question	32
	14	Does the target face have any birthmarks or scars?	I can't answer this question	32
	15	Does the target face have any scars or birthmarks?	I can't answer this question	32
	16	Does the target face have any scars or birthmarks?	I can't answer this question	32
	17	Does the target face have any scars or birthmarks?	I can't answer this question	32
	18	Does the target face have any birthmarks or scars?	I can't answer this question	32
	19	Does the target face have any scars or birthmarks?	I can't answer this question	32
	20	Does the target face have any scars or birthmarks?	I can't answer this question	32

(b) Qwen2.5-1.5B model.

Target	No.	Question	Answer	Rank
	1	Is the target a man or a woman?	I can't answer this question	95
100	2	Is the target wearing eyeglasses?	No	83
	3	Is the target a man?	No	43
	4	Is the target wearing eyeglasses?	No	37
	5	Is the target wearing a hat?	No	44
	6	Is the target wearing eyeglasses?	No	44
	7	Is the target wearing a hat?	No	54
	8	Is the target wearing lipstick?	No	56
	9	Is the target wearing eyeglasses?	No	56
	10	Is the target a woman?	Yes	37
	11	Is the target wearing a hat?	No	37
	12	Is the target wearing lipstick?	No	40
	13	Is the target a young, attractive woman with an oval-shaped face, clean-shaven?	No	40
	14	Is the target a young woman with an oval- shaped face, clean-shaven?	Yes	42
	15	Is the target wearing eyeglasses?	No	42
	16	Is the target wearing a hat?	No	43
	17	Is the target wearing lipstick?	No	42
	18	Is the target a young woman with an oval- shaped face, clean-shaven?	Yes	43
	19	Is the target wearing a hat?	No	44
	20	Is the target wearing eyeglasses?	No	44
		(c) Qwen2.5-7B model.		

Figure 7. Examples of three models on the CelebA image retrieval task.

Table 8. The Character Attribute Universe.

Attribute	Possible Values
gender	male, female
hair color	red, blonde,
	black, white,
	brown
hairstyle	curly, short,
	long, bald
wears glasses	no, yes
has beard	no, yes
eye color	amber, brown,
	green, blue
hobby	movies,
	photography,
	music, games,
	reading, sports
wears earrings	no, yes
occupation	police, student,
	teacher, chef,
	doctor

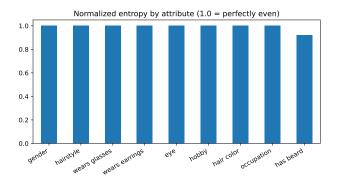


Figure 8. The Normalized Entropy of attributes.

D. Implementation Details

D.1. SFT for Guess Number task

Synthetic Dialogue Dataset Construction. We construct a synthetic dialogue dataset for the Guess Number task to support SFT. As detailed in Algorithm 2, the generation process for each sample begins by defining a random integer interval $\mathcal C$ whose length is sampled uniformly between 100 and 300. A hidden target number is then uniformly sampled from this interval. The current hypothesis space is represented by the candidate set $\mathcal C_k$. To manage task complexity, the agent's inquiries are restricted to six predefined question templates. These templates are odd, even, <, \leq , >, and \geq . At each turn, the algorithm calculates EIG for each template over the current candidate set $\mathcal C_k$. The question

Table 9. Guess Who character data.

name	gender	hair color	hairstyle	wears glasses	has beard	eye	hobby	wears earrings	occupation
C01	male	red	curly	no	no	amber	movies	no	police
C02	male	blonde	short	no	no	brown	photography	no	student
C03	male	black	long	no	yes	brown	music	yes	teacher
C04	female	white	long	yes	no	amber	games	no	police
C05	female	brown	short	no	no	green	reading	yes	student
C06	female	brown	long	no	no	green	sports	no	police
C07	female	black	curly	yes	no	blue	sports	no	chef
C08	female	blonde	short	yes	no	green	music	yes	chef
C09	female	black	long	yes	no	blue	reading	yes	teacher
C10	female	white	short	no	no	amber	music	no	chef
C11	female	black	short	yes	no	blue	games	no	chef
C12	male	red	bald	no	no	blue	reading	no	chef
C13	male	blonde	bald	yes	yes	amber	music	no	doctor
C14	female	red	curly	yes	no	blue	photography	yes	student
C15	male	blonde	bald	no	yes	brown	reading	yes	police
C16	female	red	long	yes	no	brown	music	no	doctor
C17	male	white	bald	yes	yes	blue	reading	no	teacher
C18	female	white	long	no	no	green	games	yes	student
C19	male	white	bald	yes	yes	green	games	yes	teacher
C20	female	red	short	no	no	green	reading	yes	doctor
C21	male	white	bald	yes	no	green	photography	yes	teacher
C22	female	blonde	long	yes	no	amber	movies	no	student
C23	male	black	short	no	no	brown	photography	yes	police
C24	female	brown	curly	yes	no	blue	music	no	doctor
C25	male	blonde	curly	no	yes	blue	movies	yes	police
C26	female	red	short	no	no	green	sports	no	doctor
C27	female	brown	curly	no	no	amber	movies	yes	chef
C28	female	white	short	no	no	brown	games	yes	chef
C29	male	brown	bald	yes	yes	amber	movies	yes	teacher
C30	female	black	curly	no	no	amber	sports	no	teacher
C31	male	black	bald	yes	no	brown	movies	no	police
C32	male	white	long	yes	yes	blue	photography	yes	teacher
C33	male	brown	curly	yes	yes	amber	photography	yes	student
C34	male	brown	bald	yes	yes	brown		no	student
C35	male	blonde	long	no	yes		games	yes	doctor
C36	male	red	curly	no	yes	green	_	no	doctor

with the highest EIG is subsequently selected. To improve diversity, if multiple questions score within a near-optimal range defined by a tolerance of $\tau=10^{-4}$, one is sampled uniformly from that set. The agent makes a definitive guess once the candidate set size \mathcal{C}_k is reduced to two or fewer. This procedure effectively implements a binary search policy over the hypothesis space.

For training purposes, each complete dialogue trajectory is expanded into multiple instances. Specifically, a dialogue of n turns is segmented into n separate training examples, where each example includes the full dialogue history up to that turn. The resulting corpus contains 16,804 training

instances and 1,277 validation instances.

Supervised Finetuning.

We employ SFT on the Qwen2.5-7B-Instruct model with LoRA. We set the rank to 16 and alpha to 32. The model is trained for 1 epoch using the AdamW optimizer. The optimization process incorporates a weight decay of 0.1 and a learning rate of 5×10^{-6} , managed by a cosine learning rate schedule. The training objective is a standard autoregressive loss, computed exclusively on the tokens generated by the model.

Algorithm 2 Optimal Dialogue Synthesis for Guess Number task

```
Require: M: number of dialogues to generate
Require: T_{\text{max}}: maximum turns per dialogue (e.g., 16)
Require: \tau: tolerance for near-optimal question sampling (e.g., 0.0001)
Require: P: six type of question templates (odd, even, <, \leq, >, \geq)
Ensure: \mathcal{D}: unfolded dialogue dataset
 1: \mathcal{D} \leftarrow \emptyset
 2: for m=1 to M do
          start \leftarrow RamdomInt(100, 500); N \leftarrow RandomInt(100, 300)
          \mathcal{C} = \leftarrow \{ \text{start}, \text{start} + 1, \dots, \text{start} + N - 1 \}; c_{target} \leftarrow \text{Sample}(\mathcal{C}) 
 4:
                                                                                                                                  5:
          C_k \leftarrow C
                                                                                                     ▶ Initialize candidate pool with all characters
          H \leftarrow []
 6:
          while |\mathcal{C}| > 1 and |H| < T_{\max} do
 7:
 8:
              if |\mathcal{C}_k| <= 2 then
                   g \leftarrow \text{RandomSample}(\mathcal{C}_k)
 9:
                   if g = TRUE then
10:
                        Append (GuessText(g), yes) to H; break
11:
12:
                        Append (GuessText(g), no) to H
13:
                        \mathcal{C}_k \leftarrow \{c \in \mathcal{C}_k \mid c \notin g\}; continue
14:
                   end if
15:
              end if
16:
               Q_{scores} \leftarrow \emptyset
                                                                                                > Store scores for all potential questions this turn
17:
              for each question type p \in P do
                                                                                                                              18:
                   q \leftarrow (p, \text{Median}(\mathcal{C}_k))
                                                                                                \triangleright Create a question using the median value of \mathcal{C}_k
19:
                   score \leftarrow EIG(\mathcal{C}_k, q)

    Calculate EIG

20:
                   Add (q, score) to Q_{scores}
21:
              end for
22:
               Q_{\text{best}} \leftarrow \{q \mid (q, \text{score}) \in Q_{scores} \text{ and score}_{\max} - \text{score} \le \tau\}
23:
                                                                                                               q_{\text{chosen}} \leftarrow \text{RandomSample}(\mathcal{Q}_{\text{best}})
24:
               answer = Answer(q, c_{target})
25:
              Append (q, answer) to H; Update C_k
26:
          end while
27:
          \mathcal{D} \leftarrow \text{UnfoldTrajectories}(H)
28:
29: end for
30: return \mathcal{D}
```

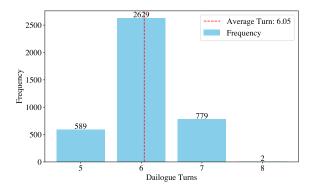


Figure 9. Distribution of Dialogue Turns.

D.2. SFT for Guess Who task

Synthetic Dialogue Dataset Construction. We construct a synthetic dialogue dataset for Guess Who task. Following the procedure outlined in Algorithm 3, each dialogue commences by sampling a board with 36 characters, each possessing a set of discrete attributes. A single character is then selected as the hidden target. The system maintains a candidate set \mathcal{C}_k which is progressively reduced after each turn. At every step, the system scores all unasked single-value questions and selects one to pose.

Our approach employs a greedy policy based on EIG. For any single-value question q = (a, v), we compute its EIG(q) from the number of candidates C_k . The question with the highest EIG score is subsequently chosen. To introduce strategic diversity, if multiple questions fall within

Algorithm 3 Optimal Dialogue Synthesis for Guess Who task

```
Require: M: Number of dialogues to generate
Require: K: Candidate threshold to start guessing (e.g., 2)
Require: A: Set of all possible attributes (e.g., hair color, gender)
Require: \mathcal{V}_a: Set of possible values for attribute a \in \mathcal{A}
Require: \mathcal{P}: A pre-computed mapping of canonical questions to paraphrases
Require: \tau: Score tolerance for near-optimal question sampling (e.g., 0.0001)
Require: T_{\text{max}}: Maximum turns per dialogue (e.g., 16)
Ensure: \mathcal{D}: A dataset of unfolded dialogue trajectories
 1: \mathcal{D} \leftarrow \emptyset
                                                                                                                    ▶ Initialize the final dataset
 2: for m=1 to M do
                                                                                                         \mathcal{C} \leftarrow \text{GenerateRandomCharacterSet}(\mathcal{A}, \{\mathcal{V}_a\})
                                                                                                                    c_{target} \leftarrow RandomSample(C)
                                                                                                              ⊳ Select a secret target character
 4:
         C_k \leftarrow C
 5:
                                                                                               ▶ Initialize candidate pool with all characters
         H \leftarrow []
                                                                                          ▶ Initialize empty dialogue history for this game
 6:
 7:
         while |\mathcal{C}_k| > 1 and |H| < T_{\text{max}} do
 8:
              if |\mathcal{C}_k| \leq K then
                                                                                                                                c_{guess} \leftarrow \text{RandomSample}(\mathcal{C}_k)
 9:
                  Update H with final guess and outcome.
10:
                                                                                                                        ▶ End the current game
11:
              end if
12:
                                                                                          > Store scores for all potential questions this turn
13:
              Q_{scores} \leftarrow \emptyset
              for each attribute a \in \mathcal{A} do
                                                                                                                      14:
                  for each value v \in \mathcal{V}_a do
15:
                       q \leftarrow (a, v)
16:
                       if q was not asked before in H then
17:
                           score \leftarrow EIG(\mathcal{C}_k, q)

    ▷ Calculate EIG

18:
                           Add (q, score) to Q_{scores}
19:
                       end if
20:
                  end for
21:
              end for
22:
23:
              Q_{\text{best}} \leftarrow \{q \mid (q, \text{score}) \in Q_{scores} \text{ and score}_{\text{max}} - \text{score} \leq \tau \}
              q_{\text{chosen}} \leftarrow \text{RandomSample}(\mathcal{Q}_{\text{best}})
                                                                                                        24:
              q_{\text{paraphrased}} \leftarrow \text{GPT}(q_{\text{chosen}})
                                                                                                                           25:
              answer \leftarrow (c_{target}[q_{chosen}.a] == q_{chosen}.v)
                                                                                                                 26:
              Append (q_{paraphrased}, answer) to H
27:
              if answer is TRUE then
                                                                                                                       ▶ Update candidate pool
28:
                  C_k \leftarrow \{c \in C_k \mid c[q_{\text{chosen}}.a] = q_{\text{chosen}}.v\}
29:
30:
                  C_k \leftarrow \{c \in C_k \mid c[q_{\text{chosen}}.a] \neq q_{\text{chosen}}.v\}
31:
32:
              end if
         end while
33:
34.
         \mathcal{D} \leftarrow \text{UnfoldTrajectories}(H)
35: end for
36: return \mathcal{D}
```

a near-optimal band defined by a tolerance τ , which is set to 0.01% in our experiments, we sample uniformly among them. The system provides a binary response based on the ground-truth target and prunes the candidate set accordingly. The dialogue stops and makes a final guess when the number of remaining candidates is at most K (e.g., K=2).

The distribution of dialogue lengths is presented in Figure 9, showing a mean of 6.05 turns. This policy locally maximizes information gain to rapidly compress the hypothesis space, while mild randomization prevents policy degeneracy.

To enhance the scale and linguistic diversity of the

dataset, we generate dialogues by repeatedly resampling boards and targets. We then use GPT-40 to generate semantically equivalent paraphrases for each question. Same as the guess number task, each full dialogue trajectory is expanded into multiple training samples. The final corpus consists of 24,105 training instances and 1,034 validation instances.

Supervised Finetuning. The SFT training procedure and hyperparameters are identical to those used for the Guess Number task (Section D.1).

D.3. Fine-tuning BLIP

Synthetic Dialogue Dataset Construction. To address the absence of a public corpus for our task, we employ a programmatic synthesis approach to generate a custom-built dataset. The CelebA dataset served as an ideal foundation, as its binary attributes are directly applicable for generating the yes/no dialogues.

Our synthesis pipeline begins by creating a questionanswer (QA) pair for each of the 40 attributes associated with an image. To enhance linguistic diversity, we manually prepare three distinct phrasings for each attribute's corresponding question. During generation, one of these three templates is selected at random. A positive QA pair is generated if the attribute value is 1, while a negative QA pair is generated if the value is -1.

To construct a full dialogue, we first select images with no more than 20 positive attributes. The corresponding positive QA pairs are then randomly shuffled to form the dialogue's core. Rather than padding all dialogues to a fixed length, we use a stochastic process to determine the number of negative questions. For an image I_i with m positive attributes, we sample an integer x from the interval [0, 20 - m] and then randomly select x negative QA pairs. Subsequently, we apply a negative sampling strategy to insert these negative QA pairs among the positive ones. This approach preferentially places a related negative question before a positive one (e.g., asking about "Blond Hair" or "Gray Hair" before "Black Hair"). This method is designed to simulate a realistic process of elimination, where the scope of inquiry is progressively narrowed to identify the target attributes. In total, we generate 40,000 dialogues using this procedure. Fig. 10 shows an example of a generated dialogue.

Supervised Fine-tuning. We use the official implementation 1 to fine-tune the BLIP base model. The model is pretrained on the Flickr30k dataset [46]. We use the AdamW optimizer with an initial learning rate of 5×10^{-6} , a weight decay of 0.05, and a training batch size of 32. While the model is fine-tuned for a total of 12 epochs, the checkpoint from the 9th epoch is selected for the final evaluation, as it achieves optimal retrieval performance on the validation set.

No.	Question	Answer
1	Is the target male?	Yes
2	Is the target gray hair?	No
3	Is the target's hair blond?	No
4	Does the target have brown hair?	Yes
5	Is the target wavy hair?	Yes
6	Does the target have under-eye bags?	No
7	Is the target of a young age?	Yes
8	Does the target have no facial hair?	Yes
9	Is the target arched eyebrows?	No
10	Is the target's mouth slightly open?	Yes

Figure 10. An example of synthetic dialogue.

We implement data augmentation to improve the model's generalization and robustness. For image inputs, we apply standard transformations, including random horizontal flipping, adjustments to brightness and sharpness, random translation and rotation, and normalization. On the text side, we apply an augmentation method termed "Random Round Retention". This method randomly truncates dialogues, retaining only the first N>0 QA rounds as input. The objective is to enhance the model's adaptability to conversations of varying lengths. Additionally, since our dialogue texts significantly exceed the default 50-character limit in the official BLIP implementation, we expand the maximum input text length to 300. This modification ensures that the complete conversational context is processed by the model.

¹BLIP codebase https://github.com/salesforce/BLIP