Memory-QA: Answering Recall Questions Based on Multimodal Memories

Hongda Jiang[*†], **Xinyuan Zhang**[*†], **Siddhant Garg, Rishab Arora,**
Shiun-Zu Kuo, Jiayang Xu, Aaron Colak, Xin Luna Dong[†]
Meta Reality Labs
[†]{jhd,dylanz426,lunadong}@meta.com

Abstract

We introduce Memory-QA, a novel real-world task that involves answering recall questions about visual content from previously stored multimodal memories. This task poses unique challenges, including the creation of task-oriented memories, the effective utilization of temporal and location information within memories, and the ability to draw upon multiple memories to answer a recall question. To address these challenges, we propose a comprehensive pipeline, PENSIEVE , integrating memory-specific augmentation, time- and location-aware multi-signal retrieval, and multi-memory QA fine-tuning. We created a multimodal benchmark to illustrate various real challenges in this task, and show the superior performance of PENSIEVE over state-of-the-art solutions (up to 14% on QA accuracy).

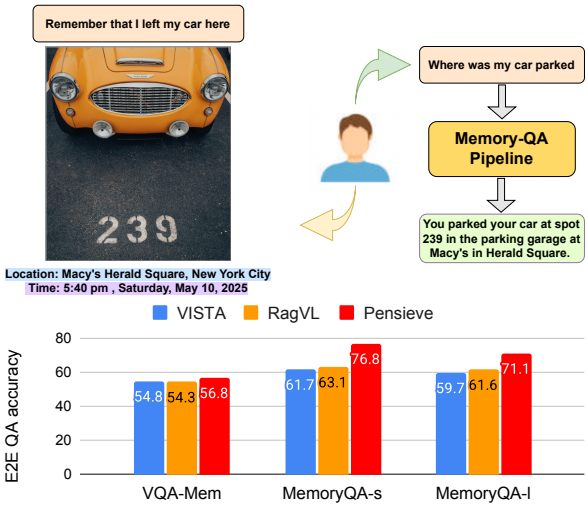


Figure 1: A Memory-QA example. Our PENSIEVE solution improves end-to-end QA accuracy over state-of-the-art MM-RAG systems (Zhou et al., 2024; Chen et al., 2024) by up to 14%.

1 Introduction

Envision a smart personal assistant capable of persistently remembering events from an individual's life—under explicit user permission—and answer recall questions like *"Where did I park my car?" "I had some very good Korean hotpot a while back but which restaurant was that?" "Does this skirt have lower price than the similar one I saw at Macy's yesterday?"* This vision dates back to Vannevar Bush's seminal concept of *MEMEX (MEMory & EXpansion)* (Bush et al., 1945), and has recently been revitalized under the emerging paradigm of the *Second Brain* (Forte, 2022).

Achieving this vision introduces a number of technical challenges, including memory recording, compression, storage, and search. In this paper, we take an initial step towards this long-term vision by addressing a simpler yet essential sub-problem: *recording memory snapshots on user request and enabling question answering over these recorded memories*. For instance, a user may issue invocation commands such as *"remember my parking lot," "remember this restaurant", "remember this dress"*, or more generally, *"remember this"* through wearable devices, mobile phones, etc. In response, the assistant captures the user's intent along with a visual snapshot of their current view. These on-demand, user-initiated memory recordings serve as the foundation for answering future memory-related questions. We refer to this task as *Memory-QA*, comprised of multimodal memory recording, memory retrieval, and question answering.

The rapid advancement in Vision-Language Models (VLM) has significantly enhanced the capabilities of intelligent assistants in understanding and reasoning over both textual and visual inputs, leading to substantial improvements in *Visual Question Answering (VQA)* (Yang et al., 2022; Zhang et al., 2023). Building on this progress, recent research on *Multi-Modal Retrieval-Augmented Generation (MM-RAG)* (Zhou et al., 2024; Chen et al.,

[*]These authors contributed equally to this work.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 24256–24278
November 4-9, 2025

2024) further extends these capabilities by first retrieving relevant images from large corpora, and then applying VQA techniques to the retrieved content. At first glance, the Memory-QA task appears to align closely with MM-RAG. However, existing MM-RAG approaches still face several unique challenges when applied to Memory-QA scenarios.

To begin with, memory-related questions are often anchored to vague temporal or spatial references, such as *"yesterday"* or *"last month"* for time, and *"at Macy's"* or *"in downtown"* for location. Effectively leveraging such anchors is crucial for accurate question answering. Moreover, many recall questions require aggregating information from multiple memory entries. For example, answering *"where did I park?"* typically involves retrieving the most recent parking memory, whereas *"what's on my shopping list?"* may require combining several past entries issued with *"remember to buy this"*. Finally, most existing VLMs are constrained by limited visual context windows, which hinders their ability to reason over a large set of multimodal memory snapshots.

In this paper, we propose PENSIEVE , the first end-to-end solution to the Memory-QA problem, grounded in three key intuitions. First, unlike standard MM-RAG tasks where the retrieval corpus is typically public and external, a personal memory repository resides in a personal context and can be explicitly augmented to enhance memory retention and retrieval. In the offline stage, we enrich each memory image with image captions generated by a Large Language Model (LLM), text extracted via Optical Character Recognition (OCR), and contextual metadata such as timestamps and geolocation. During memory retrieval, we propose a multi-signal retriever stack that incorporates temporal and location matching signals inferred from the user question. This dual-modality and condition-aware retrieval mechanism ensures more accurate and context-relevant memory selection.

Second, in contrast to general visual question types, such as object counting, spatial reasoning, activity recognition, and commonsense inference, recall questions tend to center around object and event tracking, which introduces opportunities for targeted optimization. To this end, we employ a few-shot learning image captioning module to better serve memory-related queries by predicting plausible recall questions for a given image, and incorporating the appropriate level of details in the generated captions to support such questions. Furthermore, we introduce a temporal query rewriter and fine-tune the answer generator at different stages to better align with the specific characteristics of Memory-QA.

Third, to enhance robustness and generalization, we employ multi-task instruction fine-tuning with noise injection, allowing the models to effectively handle the ambiguity and variability inherent in the retrieved memory candidates. At the question answering stage, we mitigate the challenge posed by limited context windows in VLMs by relying solely on the rich textual information generated during the memory augmentation phase. This design allows the system to reason across multiple memory snapshots without directly encoding raw images or large visual feature sets into the model's context.

In summary, our contributions are:

1. We formally define the Memory-QA problem, capturing key elements in real scenarios. We create a benchmark MemoryQA with 9,357 recall questions to illustrate real challenges.
2. We design the PENSIEVE system for end-to-end memory-QA, improving quality with memory-specific augmentation, temporal-and-location-aware multi-signal retrieval, and fine-tuned multi-memory QA.
3. We conduct extensive experiments, showing PENSIEVE improves over state-of-the-art MM-RAG solutions by up to 14% on the MemoryQA benchmark. With PENSIEVE , text-based LLMs obtain comparable results to VLMs, demonstrating a pathway to lower-cost Memory-QA.

2 Related Work

Multimodal Question Answering requires reasoning on information from diverse modalities to answer questions. It has evolved from early tasks focusing on vanilla VQA (Antol et al., 2015), to later expanding to more complex scenarios such as ManyModelQA (Hannan et al., 2020) and MultiModalQA (Talmor et al., 2021) which consider multiple modalities, as well as WebQA (Chang et al., 2022) and SnapNTell (Qiu et al., 2024) involving real-world, knowledge-seeking questions. To integrate these modalities, recent research has employed various approaches, including joint embedding of multiple modalities (Li et al., 2022; Yu et al., 2023), fusing multimodal information with LLMs (Zhang et al., 2023; Yang et al., 2023; Nan et al., 2024), and transforming all modalities into

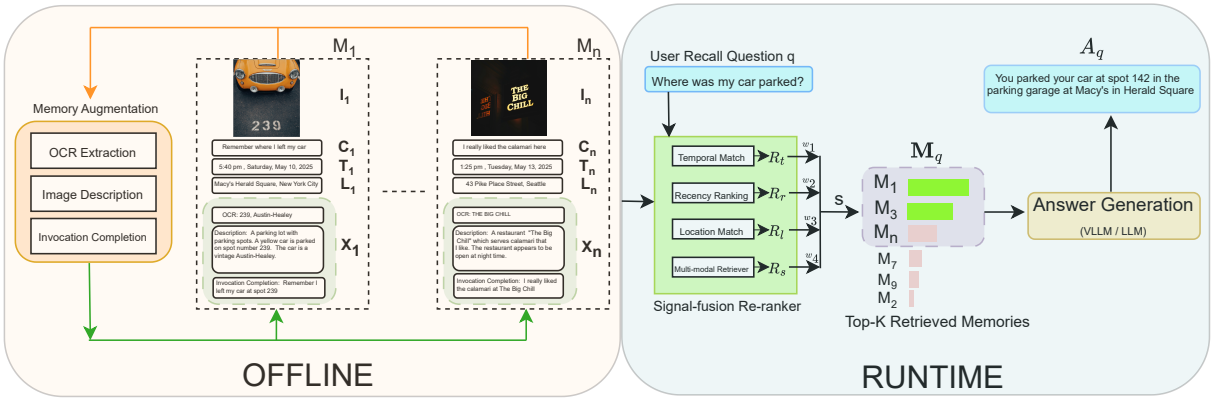


Figure 2: Our proposed pipeline PENSIEVE for Memory-QA.

text through captioning or description (Lin et al., 2022; Yang et al., 2022). Memory-QA differs from these works by focusing on recall questions to track events from previously created memories, introducing targeted optimizations.

Multimodal Retrieval-Augmented Generation has been shown to enhance both the understanding and generation capabilities of vision models (Zheng et al., 2025). Pioneering works such as MuRag (Chen et al., 2022), REACT (Liu et al., 2023c) and RA-VQA (Lin and Byrne, 2022) introduce language augmentations by parsing multimodal documents into text, aiming to improve retrieval and answer generation performance. Recent studies have further refined this approach by enhancing model reasoning capabilities (Liu et al., 2023a; Tan et al., 2024) and robustness (Chen et al., 2024; Long et al., 2025). Another line of research has explored utilizing complete vision information by directly encoding images for retrieval and generating outputs solely based on visual content (Zhou et al., 2024; Ma et al., 2024; Faysse et al., 2024; Yu et al., 2024). However, these approaches fall short in addressing the temporal and location references inherent in Memory-QA.

Vision-Language Models have been extensively explored in recent years to enhance fine-grained multimodal understanding. Building on the foundation laid by CLIP (Radford et al., 2021), which enabled contrastive visual-text alignment, researchers have developed unified models that integrate pre-trained LLMs and vision encoders such as Flamingo (Alayrac et al., 2022) and BLIP-2 (Li et al., 2023a). Additionally, works like LLaVA (Liu et al., 2023b), mPLUG-Owl (Ye et al., 2023), and MiniGPT-4 (Zhu et al., 2023) have focused on fine-tuning LLMs for better visual feature alignment. More recently, models like Llama3.2 (Grattafiori et al., 2024) and Qwen2-VL (Wang et al., 2024) combine these techniques to achieve state-of-the-art open-source performances. These works have further enabled notable improvements in vision-language tasks including VQA (Hu et al., 2024) and OCR (Shenoy et al., 2024). While some works have begun exploring multi-image understanding (Li et al., 2024a), handling large numbers of images or similar visual content remains challenging.

3 Overview

3.1 Memory-QA Definition

We now formally define the *Memory-QA* problem. Consider a repository of *memory entries* $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$. Each memory entry is a tuple $M_i = (I_i, C_i, T_i, L_i)$, where I_i denotes the image snapshot, C_i denotes the invocation command for memory recording, T_i denotes the timestamp of the memory, and L_i denotes the location where the memory is captured. An invocation command, such as "remember this dress", often provides context information such as the reason for capturing the memory or the focus area in the view, critical for later recall; it is possible that the *invocation command* does not provide extra clues, such as a general command "remember this".

The *Memory-QA* problem takes as input a recall question q asked at timestamp T_q and generates an answer according to memories in $\mathcal{M}$. A good answer shall reflect information from all memories in $\mathcal{M}$ that are relevant to the input question q.

3.2 Overview of PENSIEVE

As depicted in Figure 2, PENSIEVE consists of two parts: *offline augmentation* and *runtime QA*. Offline

augmentation expands each memory entry with auxiliary text including OCR, image descriptions, and invocation completions that enrich the invocation commands (e.g., completing *"remember where I parked"* with *"remember I parked at slot 142"*). After the augmentation the memory entry becomes $M_i^a = (I_i, C_i, T_i, L_i, X_i)$, where X_i denotes the augmented auxiliary text. Runtime QA takes user question q and proceeds in two steps, similar to MM-RAG: first, it retrieves relevant memory candidates, denoted by $\mathbf{M}_q$; then, it generates the answer A_q based on the retrieved memories.

Our solution incorporates three innovations. First, we introduce the task-oriented augmentation step for better memory retention. Second, we design memory-specific solutions for each step, including memory recording, memory retrieval, and answer generation, to optimize for Memory-QA. Third, our answer generation step is good at identifying and leveraging the set of memories that are necessary for answering the input question.

4 Methodology

We now present PENSIEVE in detail, highlighting how we leverage our three intuitions mentioned in Section 1 to address the challenges in Memory-QA.

4.1 Memory Augmentation

Offline memory augmentation takes as input a memory entry $M_i = (I_i, C_i, T_i, L_i)$, containing the snapshot image, the invocation command (text), the timestamp, and the location, and augments it with auxiliary memory clue X_i, which describes the memory snapshot in text.

To facilitate subsequent runtime recall including retrieval and answer generation (**Intuition 1: Augmentation**), we create comprehensive memory clues by leveraging the invocation command C_i and also predict potential memory-questions on the snapshot to generate the most effective memory clues (**Intuition 2: Memory-specific**).

Augmentation and encoding Our auxiliary memory clue X_i contains three fields: *OCR results, image caption,* and *invocation completion*, stored and indexed separately. First, since texts in the view are often critical in answering recall questions, such as *restaurant name, product price, name card,* and *phone number on a poster*, we apply OCR models to extract textual information from the memory image. Second, we leverage VLMs to generate detailed description of the image, to provide a foundation for understanding the visual content. Third, we complete the invocation command with the information in the image; taking *"remember this restaurant"* as an example, the invocation completion can be "remember this Korean restaurant named Kochi". We do so by invoking VLMs to generate the complete command.

We use a multimodal encoder to embed the image and text concatenations. Formally,

$$\mathbf{M}_i = \mathcal{F}(I_i, C_i, X_i, L_i) \in \mathcal{R}^d,$$

where $\mathcal{F}(\cdot)$ denotes the multimodal encoder, with d being the embedding size.

QA-guided Image Description Generation Given that Memory-QA focuses on task-oriented questions with the intention of recalling useful information, we can enhance vanilla image captioning to generate descriptions specifically effective in answering a wide range of memory recall questions posed on the memory entry. The QA-guided image description generation is achieved through a few-shot learning approach using VLMs.

Starting with example task-oriented questions, we prompt a VLM to generate potential recall questions on the image and the answers to the questions; we use in-context learning and provide a diverse list of recall questions as few-shot examples. We then prompt the model to generate a comprehensive image description that enables answering the recall questions without referring to the image. The resulting model will be able to focus on salient, question-relevant image features for captioning. The generated QA-guided image descriptions are used as a crucial memory element for retrieval and answer generation.

4.2 Memory Retrieval

The runtime multimodal retrieval step takes as input a recall question q and the augmented memory repository $\mathcal{M}^a$, identifies a set of memory entries $\mathcal{M}_q^a$ that are relevant to the question.

To effectively utilize all memory components, we employ a multi-signal retriever that integrates information from snapshot images, invocation commands, temporal/location context, and offline-generated augmentations. Our retrieval stack first runs a temporal and location matching module in parallel with a multimodal retriever. The matching module computes date match scores, location

match scores, and recency scores, while the multimodal retriever predicts similarity scores. These independent signals are then combined by a re-ranker to produce the final ranked list of memories.

Temporal / location matching module We first employ an LLM-based date parser to extract temporal information from the raw query, specifically the search date range (T_s, T_e) and a boolean B_r indicating whether recent memories should be favored. This extracted information is used to calculate date match and recency scores as follows.

For queries with a non-empty search date range (e.g., *"What did I save last week?"*), the date match score R_t for memory M_i is computed as:

$$R_t(M_i, q, T_q) = \mathbf{1}\{T_s(q, T_q) \leq T_i \leq T_e(q, T_q)\},$$

where $\mathbf{1}\cdot$ denotes an indicator function.

For queries seeking recent memories (e.g., *"Where did I park last time?"*), the recency score R_r for memory M_i is calculated as:

$$R_r(M_i, T_q) = B_r\left(e^{-\frac{\delta}{Q_\mathrm{s}}} + e^{-\frac{\delta}{Q_\mathrm{m}}} + e^{-\frac{\delta}{Q_\mathrm{l}}}\right)/3,$$

where $\delta = T_q - T_i$ represents the interval between memory creation and the query time. The constants $Q_\mathrm{s} = 3$ days, $Q_\mathrm{m} = 90$ days, $Q_\mathrm{l} = 365$ days, simulate varying rates of memory decay over short/middle/long time periods (Li et al., 2023b). Note that if the parser predicts $B_r = 0$, all recency scores default to zero.

To prioritize memories whose creation locations match the user's query, we compute a location match score R_l using the BM25 algorithm:

$$R_l(M_i, q) = \mathrm{BM25}(L_i, q).$$

Multimodal retriever We use a multimodal encoder to embed both memory content and the query. The similarity score R_s is defined as the dot product between these embeddings:

$$R_s(M_i^a, q) = \mathbf{M}_i^\top \cdot \mathcal{F}(q),$$

where $\mathcal{F}(\cdot)$ represents the multimodal encoder. $\mathbf{M}_i = \mathcal{F}(M_i^a)$ is the memory embedding produced by the same encoder during offline encoding.

Signal fusion re-ranker: Our re-ranker integrates the temporal/location matching signals and the multimodal retriever signal. For a given query q, the final retrieval score s_i for memory candidate M_i is computed as a weighted sum:

$$\begin{aligned} s_i = {} & w_t R_t(M_i, q, T_q) + w_r R_r(M_i, T_q) \\ & + w_l R_l(M_i, q) + w_s R_s(M_i^a, q), \end{aligned}$$

where w_t, w_r, w_l, and w_s are weights for each signal. To enable domain-specific customization, we optimize the retriever weights by training a linear model on a small volume of domain data (see Appendix A.5). This approach allows us to adapt the model to the specific memory domain while providing additional model interpretability through the learned weights. In the end, we send the top-K candidates for answer generation: $\mathcal{M}_q^a = \mathrm{TopK}\ \{s(\mathcal{M}^a, q)\}$.

4.3 Answer Generation

The runtime answer generation step takes as input a recall question q and the retrieved memory set $\mathcal{M}_q^a \subset \mathcal{M}^a$, and generates the answer A_q to the question, $A_q = Gen(q, \mathcal{M}_q^a)$. In this work, leveraging the high-quality memory augmentations, we argue that only using the rich textual memories to generate A_q with a text-based LLM is not only lower-cost, but also achieves comparable performances as using multimodal memories.

The main challenge for this step is that answer generation may need to aggregate information from multiple sources. We conduct fine-tuning such that the VQA model can effectively identify positive and negative candidates from $\mathbf{M}_q$ and answer the question only based on relevant memories (**Intuition 3: Multi-memory QA**) .

Noise-injected instruction tuning To mitigate the negative impact of irrelevant memories retrieved, we employ noise-injected training (Chen et al., 2024). This approach creates the training dataset by including up to 2 confusing candidates as negative examples, which are presented alongside positive memories in a similar manner as mentioned above. By doing so, the fine-tuned model is trained to robustly discern relevant from irrelevant information, thereby strengthening its memory comprehension and ability to filter out noise.

Multi-task instruction tuning Considering the nature of this answer generator is to do two tasks at the same time: detect positive candidates from $\mathbf{M}_q$ and generate an answer A_q to the question based on relevant candidates, we propose multi-task fine-tuning to jointly train two tasks simultaneously and improve the answer correspondence with relevant memories. Specifically, the LLM outputs *a list of positive memory Ids followed by the generated answer*. Training aims to optimize a standard autoregressive cross-entropy loss

Datasets	#Images	#Samples	Recall Question Only	Time & Location
MemoryQA				
train	3,011	6,386	Yes	Yes
test-s	189	1,326	Yes	Yes
test-l	2,789	2,971	Yes	Yes
VQA-Mem	1,469	1,811	Yes	No
WebQA	39,000	2,511	No	No

Table 1: Statistics of the datasets used in this work.

$L = -\sum_{t=1}^{T} \log P(y_t|y_{<t})$ computed over the entire response, where T is the total sequence length.

5 Experiment Setup

5.1 Datasets

We experiment with three benchmarks: our in-house dataset MemoryQA[1], an extended version of VQA (Antol et al., 2015), called VQA-Mem, and WebQA (Chang et al., 2022) (statistics shown in Table 1). All datasets are formatted as positive and negative samples for each question, with VQA-Mem's negative samples being randomly selected to synthesize user memory diversity.

MemoryQA Our in-house benchmark, MemoryQA, comprises 5,800 images captured from daily life using wearable devices such as smart glasses. What sets MemoryQA apart from other multimodal QA benchmarks is the inclusion of temporal and location information for each image. For the test sets, we employ human annotators to craft invocation commands and timestamped recall questions for each image, and also to verify the accuracy of each answer (see Appendix A.3.2 for more details). In contrast, the training set was generated using VLM without human annotations. We have two test sets with different sizes, MemoryQA-s, and MemoryQA-l with more challenging questions and more diverse image types.

VQA-Mem To adapt VQA for the Memory-QA task, we first prompt Llama3.3-70B to remove QA pairs that do not pertain to recall questions and answers, and only keep images with at least one recall QA pair, ensuring relevance for Memory-QA. Then, for each selected image, we use Llama3.2-90B to generate two different invocation commands that trigger memory recording.

WebQA We utilize the image modality data from the WebQA validation set. We keep the original images and QA pairs without filtering, so this is more of a standard multimodal QA dataset. We treat image titles as invocation commands for our problem setting.

[1]MemoryQA dataset will be available at: https://github.com/facebookresearch/MemoryQA/

Model	WebQA	VQA-Mem	MemoryQA s	MemoryQA l
Baseline (w/o aug., CLIP)				
GPT-4o (vis)	64.4	54.2	58.2	49.2
Llama3.2-90B (vis)	54.0	50.9	53.1	46.9
Llama3.3-70B (txt)	13.8	6.5	27.0	28.4
SOTA MM-RAG systems, w/ GPT-4o				
VISTA	69.1	54.8	61.7	59.7
RagVL	**71.1**	54.3	63.1	61.6
PENSIEVE *: w/ aug., Multi-signal Retriever*				
GPT-4o (vis)	66.4	**56.8**	**76.8**	**71.1**
Llama3.2-90B (vis)	59.4	53.8	74.7	68.5
Llama3.3-70B (txt)	39.9	46.9	74.1	70.3

Table 2: E2E QA results A_{llm}. PENSIEVE outperforms the baseline and state-of-the-art solutions on recall questions from VQA-Mem and MemoryQA by a big margin.

Method	VQA-Mem		MemoryQA-s	
Vision Methods	A_{key}	A_{llm}	A_{key}	A_{llm}
GPT-4o	**55.8**	**56.8**	**69.4**	**76.8**
w/o augmentation	52.6	53.9	61.6	67.9
w/o MS retriever	53.2	53.8	59.8	65.9
w/o QA-guided desc.	53.4	55.2	67.9	75.6
w/o time/loc. match	-	-	64.0	69.8
Text Methods	A_{key}	A_{llm}	A_{key}	A_{llm}
Llama3.3-70B	**47.5**	**46.9**	**71.0**	**74.1**
w/o augmentation	7.3	6.4	38.5	31.7
w/o MS retriever	47.0	46.0	61.7	63.1
w/o QA-guided desc.	42.7	43.1	70.5	71.0
w/o time/loc. match	-	-	66.9	67.9

Table 3: Ablation study on both datasets shows significant improvements from our design choices.

5.2 Evaluation

Our overall metric is the *E2E QA accuracy* for the recall questions, computed as the percentage of questions that are correctly and completely answered. We compare an answer with the ground truth and decide its correctness in three ways: computing the keyword overlaps (Chang et al., 2022), LLM-as-a-judge (Li et al., 2024b) with Llama3.3-70B, and decide if the ground truth is entailed in the generated answer (Lattimer et al., 2023). We denote the accuracy computed in these metrics by A_{key}, A_{llm}, and A_{ent} respectively.

5.3 Implementation

In the implementation of PENSIEVE , we use techniques introduced in Section 4. Notably, all memory augmentations are generated once before exper-

Retriever	Reranker	R@1		R@3		R@5		nDCG@3		nDCG@5	
		s	l	s	l	s	l	s	l	s	l
Baseline retriever											
BM25	-	11.2	11.1	21.4	23.3	26.8	31.4	17.1	18.8	19.3	22.2
Dragon+	-	69.6	71.0	82.2	82.5	85.8	86.0	77.1	82.2	78.5	83.5
CLIP	-	66.2	59.3	79.4	72.8	84.4	77.1	73.8	70.3	75.9	72.0
Vis-BGE-base	-	69.2	64.8	82.2	76.2	85.8	80.1	76.9	75.0	78.4	76.5
Vis-BGE-m3	-	73.5	71.2	85.5	82.7	88.4	86.4	80.7	82.2	81.9	83.6
RagVL	VLM	71.4	74.2	85.8	85.8	88.4	88.3	80.4	85.0	81.6	85.9
Vis-BGE-m3 + temporal/location matching module + Reranker											
Multi-signal	Max	71.9	71.7	85.2	88.2	89.8	91.7	79.7	84.4	81.6	85.8
Multi-signal	Sum	80.6	77.9	91.8	91.3	94.2	94.2	87.3	89.2	88.3	90.7
Multi-signal	Learned weights	**84.3**	**80.9**	**93.5**	**92.1**	**95.5**	**95.0**	**89.8**	**91.0**	**90.6**	**92.0**
Ablation study											
w/o date match score	Learned weights	76.2	73.4	86.8	83.7	89.4	87.3	82.5	83.9	83.6	85.2
w/o recency score	Learned weights	82.6	79.1	92.8	91.0	94.8	94.3	88.7	89.6	89.6	90.7
w/o location score	Learned weights	83.5	80.3	93.1	**92.1**	95.3	94.9	89.4	90.7	90.4	91.7

Table 4: Performance comparison of different retriever and reranker configurations on MemoryQA-s and MemoryQA-l. Each question has 10 to 50 candidate memories.

imentation using Lumos (Shenoy et al., 2024) for OCR and Llama3.2-90B for image descriptions and invocation completions. We compare PENSIEVE with a baseline solution that records memories without augmentations, relying on retrieval with CLIP (Radford et al., 2021) or Dragon+ (Lin et al., 2023) embedding similarity for image retrieval. We also include state-of-the-art MM-RAG systems VISTA (Zhou et al., 2024) and RagVL (Chen et al., 2024) for comparison.

Memory	sim-p	sim-n	diff	A_{key}
Accumulative Text - Llama3.3-70B				
Invocation command	32.4	14.2	18.2	7.3
+ OCR result	38.9	17.5	21.4	23.5
+ Inv. completion	44.4	19.4	25.0	31.4
+ Image caption	53.4	26.0	27.4	42.7
+ QA-guided caption	55.1	24.5	30.6	47.5

Table 5: Memory augmentation increases embedding differences between positive and negative answers, thus increases QA quality.

6 Results

We now present comprehensive experimental results to show the performance of our PENSIEVE system, and justify our various design choices.

6.1 Overall results

Table 2 compares our PENSIEVE solution with baselines and state-of-the-art systems. We have four observations. First, our approach outperforms the current SOTA MM-RAG methods by a significant margin (+14%) on MemoryQA-s and (+10%) on MemoryQA-l, and also improves on VQA-Mem (+2%), highlighting the effectiveness of our targeted solutions for this task. Although WebQA lacks recall questions, our memory-based adaptation only slightly regress the results. Second, surprisingly, even using a text LLM (Llama3.3-70B) for answer generation, we achieve comparable QA accuracies over VLMs on MemoryQA, showing a cost-effective approach for Memory-QA. Third, we observe that across different backbone models, our solutions consistently outperform vision baselines by 19% on MemoryQA-s and 22% on MemoryQA-l. Notably, the even larger improvements (>42%) are observed with text models because of the textual augmentations. Finally, despite MemoryQA-l being a much more challenging and diverse dataset, PENSIEVE still achieves robust performances with 71% E2E QA accuracy.

6.2 Ablation Study

Table 3 shows a system ablation study on MemoryQA-s and VQA-Mem, which focus on recall questions. In sum, removing any component results in lower E2E QA results on both evaluation metrics. We observe significant improvements brought by our major design choices. Taking vision methods and the MemoryQA-s dataset as an example, (1) memory augmentation improves A_{llm} by 9%; (2) multi-signal retrieval leveraging the invocation commands, raw information about time and location, and the augmentations improves A_{llm} by 11%; (3) QA-guided description generation improves A_{llm} by 1%; and (4) time/location matching improves A_{llm} by 7%. These improvements are even more pronounced when we use text-based

Answer generator	A_{key}	A_{llm}	A_{ent}
GPT-4o (*vision*)	69.4	76.8	64.4
Llama3.2-90B (*vision*)	70.0	74.7	62.1
Llama3.3-70B (*text*)	71.0	74.1	61.1
Llama3.1-8B (*text*)	65.2	66.9	56.2
Llama3.1-8B-SFT (*text*)	68.6	72.4	61.3
w/o noise-injection	67.5	69.6	59.4
w/o multi-tasking	68.4	70.4	60.4

Table 6: Answer generation performances, highlighting similar performance between vision and text models, and effectiveness of fine-tuning.

Answer generator	precision	recall	F1
GPT-4o (*vision*)	83.5	**93.4**	88.2
Llama3.2-90B (*vision*)	85.0	90.2	87.5
Llama3.3-70B (*text*)	86.0	92.6	89.2
Llama3.1-8B (*text*)	85.6	90.8	88.1
Llama3.1-8B-SFT (*text*)	**87.8**	91.2	**89.5**

Table 7: Guided decoding performances to detect positive candidates during answer generation.

models, where the extra signals play an important role when vision references are absent.

6.3 Memory Augmentation Results

To investigate the impact of each memory augmentation component, we conduct experiments on VQA-Mem. In addition to end-to-end QA accuracy measured by A_{key}, we show embedding similarity scores between the textual memory components and the QA pairs, where higher scores for positive samples and lower scores for negative samples indicate more effective augmentation. Table 5 shows the metrics as we progressively add OCR results, invocation completions, image captions, and QA-guided captioning. As we add more augmentations, we observe the gap between embedding similarity with positive samples and negative samples gets bigger, leading to higher QA accuracy.

6.4 Multimodal Retrieval Results

We evaluate the effect of multi-signal retriever on MemoryQA, the only dataset that includes temporal and location references. We use recall@k and nDCG@k to measure how well the correct memories are retrieved and ranked among the top-k candidates. In Table 4, we compare our methods with a set of alternatives. The retrieval performances on MemoryQA-s and MemoryQA-l generally follow similar trends. First, we test a set of text-based and visual-based baselines, showing that visual-BGE-m3 models (Zhou et al., 2024) and RagVL

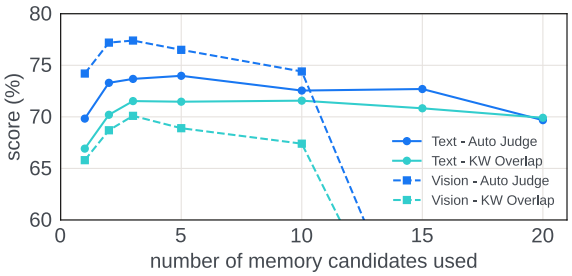


Figure 3: Generated answer quality vs number of memory candidates in the answer generator prompt.

with an VLM reranker (Chen et al., 2024) achieve the best baseline performances by leveraging both textual and visual information; however, our multi-signal retriever integrates similarity with temporal-location relevancy signals, improving various metrics by up to 13%. Second, we compare different ways to combine the various signals. In particular, using learned weights achieves a Recall@5 of 0.955 on MemoryQA-s and 0.950 on MemoryQA-l. The learned weights here are 0.08, 0.22, 0.16 and 0.53 for w_t, w_r, w_l and w_s respectively. This suggests that while the multimodal retrieving signal (w_s) dominates with half of the weights, the temporal (w_r) and location (w_l) signals takes the rest half, playing crucial roles in memory retrieval. Finally, we compared with removing the time and location signals, finding that omitting date-matching signals has the most substantial impact on performance.

6.5 Answer Generation Results

Finally, we evaluate the performance of answer generation and verify the effectiveness of our fine-tuned model on MemoryQA-s. To ensure controlled comparisons, all models are fed with the same top 3 retrieved candidates. In addition to QA accuracy (Table 6), we also compute the precision and recall of the answer generator in identifying positive candidates (Table 7). The comparisons confirm that relying solely on textual memory augmentations is an equally effective alternative to utilizing visual content for MemoryQA. In addition, it shows that fine-tuning Llama3.1-8B yields comparable QA results to 70B and achieves the best F1 score in detecting positive candidates, despite being a smaller model. Finally, omitting noise injection or multi-task instruction tuning significantly impacts the fine-tuned model's performance.

Figure 3 plots answer accuracy versus the number of retrieved candidates. Text methods get the highest accuracy with 5 retrieval candidates, and

Error Bucket	Baseline	PENSIEVE -vision	PENSIEVE -text
Correct	772 (58.2%)	1018 (76.8%)	982 (74.1%)
Retrieval error: not all positive memories are retrieved	256 (19.3%)	83 (6.3%)	82 (6.2%)
Generation error: response from wrong memories	50 (3.8%)	47 (3.5%)	49 (3.7%)
Generation error: missing key info in augmentations	N/A	90 (6.8%)	123 (9.3%)
Generation error: temporal reasoning, LLM-Judge, etc.	248 (18.7%)	88 (6.6%)	90 (6.8%)

Table 8: Error analysis on MemoryQA-s with categorized failure cases. GPT-4o backbones Baseline and PENSIEVE -vision, and Llama3.3-70B backbones PENSIEVE -text. Augmentation errors are not applicable for baselines.

Component	Config	P50 (ms)	P90 (ms)
E2E			
Runtime	w/ API (text)	1400	1950
Offline	default	1800	4710
Runtime components			
Retrieval	local & API	630	880
Query Embed	local GPU	18	19
Datetime match	API	600	850
Generate answer	API (text)	630	920
Generate answer	API (vision)	740	1470
Generate answer (fine-tuned 8B)	local GPU	920	1200
Offline components			
OCR	API	500	1200
Image caption	API	1500	4350
Memory Embed	local GPU	190	260

Table 9: Latency analysis on the proposed system. P50 and P90 denote 50th and 90th percentile latency. Here we use 40GB A100 GPU as the local GPU.

then the accuracy flattens out till reaching 20 results, showing the robustness of textual augmentations. In contrast, vision methods get highest accuracy with 3 candidates, suffer significantly when the number of memory candidates increases, because of the large visual context size.

6.6 Error Analysis

To inform future improvements, we performed a detailed analysis of model errors and categorized failure cases, as shown in Table 8. PENSIEVE substantially reduced both retrieval errors and temporal reasoning errors compared to the baseline (from 19% to 6%). The most frequent error in PENSIEVE -text is "missing key info in memories" (9.3%), which likely stems from image captioning omitting important details. In contrast, PENSIEVE -vision exhibits fewer such errors (6.8%).

6.7 Latency Analysis

Table 9 shows the latency analysis. The total estimated p50 latency for runtime QA is 1.4 seconds, which enables productionization in most AI assistants. During runtime, answer generation is the primary contributor to latency, with the vision-based approach taking 50% longer than the text-based approach at p90. This result highlights the advantage of PENSIEVE -text when its performance is competitive. In the offline stage, latency requirements are normally less stringent, and the main source of delay is image captioning.

7 Conclusion

In this paper, we introduce the Memory-QA task as a critical step towards realizing the long-standing vision of a second brain. We propose PENSIEVE , a novel end-to-end Memory-QA system integrating multimodal memory recording, memory retrieval, and answer generation. To address the challenges in Memory-QA, we propose targeted solutions including memory-specific augmentations, QA-guided image description generation, temporal and location-fused multi-signal retrieval, and multi-memory QA fine-tunings. To facilitate research in this area, we create a new multimodal QA dataset and extend existing VQA benchmarks with memory-centric recall questions tailored to this new task. We conduct extensive experiments, demonstrating the effectiveness, efficiency, and adaptability of our approach on Memory-QA.

8 Limitations

Despite the contributions of this research, there are several limitations that warrant consideration. Firstly, while the proposed approach is applicable to other multimodal QA tasks, its performance on these tasks may not be guaranteed to match the level achieved on the Memory-QA task. Secondly, to ensure fair comparisons with API-based models such as GPT-4o, we did not fine-tune large-scale LLMs or VLMs, which could have further improved the models' performances. Finally, the datetime matching module in the proposed multi-signal retriever has limitations in handling specific holidays whose dates change annually, which could impact its accuracy in certain scenarios.

9 Potential Risks and Ethical Considerations

Our work adheres to high ethical standards in data collection, usage, and publication. Below, we outline key aspects of our ethical compliance to potential risks:

- The dataset used in this study contains no offensive, harmful, or inappropriate content.
- Any PII, such as physical addresses, was synthetically generated and does not correspond to real individuals.
- All images in the Memory-QA dataset were collected with informed consent from contributors and are approved for research use. Human faces were blurred and manually verified, and any images containing unblurred or clearly identifiable faces were removed.
- The dataset does not contain, infer, or annotate any protected or sensitive attributes, such as sexual orientation, political affiliation, religious belief, or health status.
- No characteristics of the human subjects (e.g., age, gender, identity) were self-reported or inferred during data collection or analysis.
- The dataset is made available strictly for research purposes. It is not intended for commercial use or deployment in any application.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Vannevar Bush et al. 1945. As we may think. *The atlantic monthly*, 176(1):101–108.

Yingshan Chang, Mridu Narang, Hisami Suzuki, Guihong Cao, Jianfeng Gao, and Yonatan Bisk. 2022. Webqa: Multihop and multimodal qa. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16495–16504.

Wenhu Chen, Hexiang Hu, Xi Chen, Pat Verga, and William W Cohen. 2022. Murag: Multimodal retrieval-augmented generator for open question answering over images and text. *arXiv preprint arXiv:2210.02928*.

Zhanpeng Chen, Chengjin Xu, Yiyan Qi, and Jian Guo. 2024. Mllm is a strong reranker: Advancing multimodal retrieval-augmented generation via knowledge-enhanced reranking and noise-injected training. *arXiv preprint arXiv:2407.21439*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre Colombo. 2024. Colpali: Efficient document retrieval with vision language models. In *The Thirteenth International Conference on Learning Representations*.

Tiago Forte. 2022. *Building a second brain: A proven method to organize your digital life and unlock your creative potential*. Simon and Schuster.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Darryl Hannan, Akshay Jain, and Mohit Bansal. 2020. Manymodalqa: Modality disambiguation and qa over diverse inputs. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 7879–7886.

Wenbo Hu, Yifan Xu, Yi Li, Weiyue Li, Zeyuan Chen, and Zhuowen Tu. 2024. Bliva: A simple multimodal llm for better handling of text-rich visual questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 2256–2264.

Thorsten Joachims. 2002. *Learning to classify text using support vector machines*, volume 668. Springer Science & Business Media.

Barrett Lattimer, Patrick CHen, Xinyuan Zhang, and Yi Yang. 2023. Fast and accurate factual inconsistency detection over long documents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1691–1703.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Dawei Li, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu,

et al. 2024b. From generation to judgment: Opportunities and challenges of llm-as-a-judge. *arXiv preprint arXiv:2411.16594*.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR.

Yang Li, Yangyang Yu, Haohang Li, Zhi Chen, and Khaldoun Khashanah. 2023b. Tradinggpt: Multi-agent system with layered memory and distinct characters for enhanced financial trading performance. *arXiv preprint arXiv:2309.03736*.

Yongqi Li, Wenjie Li, and Liqiang Nie. 2022. Mmcoqa: Conversational question answering over text, tables, and images. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4220–4231.

Sheng-Chieh Lin, Akari Asai, Minghan Li, Barlas Oguz, Jimmy Lin, Yashar Mehdad, Wen-tau Yih, and Xilun Chen. 2023. How to train your dragon: Diverse augmentation towards generalizable dense retrieval. *arXiv preprint arXiv:2302.07452*.

Weizhe Lin and Bill Byrne. 2022. Retrieval augmented visual question answering with outside knowledge. *arXiv preprint arXiv:2210.03809*.

Yuanze Lin, Yujia Xie, Dongdong Chen, Yichong Xu, Chenguang Zhu, and Lu Yuan. 2022. Revive: Regional visual representation matters in knowledge-based visual question answering. *Advances in neural information processing systems*, 35:10560–10571.

Bingshuai Liu, Chenyang Lyu, Zijun Min, Zhanyu Wang, Jinsong Su, and Longyue Wang. 2023a. Retrieval-augmented multi-modal chain-of-thoughts reasoning for large language models. *arXiv preprint arXiv:2312.01714*.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916.

Haotian Liu, Kilho Son, Jianwei Yang, Ce Liu, Jianfeng Gao, Yong Jae Lee, and Chunyuan Li. 2023c. Learning customized visual models with retrieval-augmented knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15148–15158.

Xinwei Long, Zhiyuan Ma, Ermo Hua, Kaiyan Zhang, Biqing Qi, and Bowen Zhou. 2025. Retrieval-augmented visual question answering via built-in autoregressive search engines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 24723–24731.

Xueguang Ma, Sheng-Chieh Lin, Minghan Li, Wenhu Chen, and Jimmy Lin. 2024. Unifying multimodal retrieval via document screenshot embedding. *arXiv preprint arXiv:2406.11251*.

Linyong Nan, Weining Fang, Aylin Rasteh, Pouya Lahabi, Weijin Zou, Yilun Zhao, and Arman Cohan. 2024. Omg-qa: Building open-domain multi-modal generative question answering systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1001–1015.

Jielin Qiu, Andrea Madotto, Zhaojiang Lin, Paul A Crook, Yifan Ethan Xu, Xin Luna Dong, Christos Faloutsos, Lei Li, Babak Damavandi, and Seungwhan Moon. 2024. Snapntell: Enhancing entity-centric visual question answering with retrieval augmented multimodal llm. *arXiv preprint arXiv:2403.04735*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Ashish Shenoy, Yichao Lu, Srihari Jayakumar, Debojeet Chatterjee, Mohsen Moslehpour, Pierce Chuang, Abhay Harpale, Vikas Bhardwaj, Di Xu, Shicong Zhao, et al. 2024. Lumos: Empowering multimodal llms with scene text recognition. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5690–5700.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodal{qa}: complex question answering over text, tables and images. In *International Conference on Learning Representations*.

Cheng Tan, Jingxuan Wei, Linzhuang Sun, Zhangyang Gao, Siyuan Li, Bihui Yu, Ruifeng Guo, and Stan Z Li. 2024. Retrieval meets reasoning: Even high-school textbook knowledge benefits multimodal reasoning. *arXiv preprint arXiv:2405.20834*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Qian Yang, Qian Chen, Wen Wang, Baotian Hu, and Min Zhang. 2023. Enhancing multi-modal multi-hop question answering via structured knowledge and unified retrieval-generation. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 5223–5234.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. 2022. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 3081–3089.

Qinghao Ye, Haiyang Xu, Guohai Xu, Jiabo Ye, Ming Yan, Yiyang Zhou, Junyang Wang, Anwen Hu, Pengcheng Shi, Yaya Shi, et al. 2023.

mplug-owl: Modularization empowers large language models with multimodality. *arXiv preprint arXiv:2304.14178*.

Bowen Yu, Cheng Fu, Haiyang Yu, Fei Huang, and Yongbin Li. 2023. Unified language representation for question answering over text, tables, and images. In *Findings of the Association for Computational Linguistics*.

Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, et al. 2024. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *arXiv preprint arXiv:2410.10594*.

Le Zhang, Yihong Wu, Fengran Mo, Jian-Yun Nie, and Aishwarya Agrawal. 2023. Moqagpt: Zero-shot multi-modal open-domain question answering with large language model. *arXiv preprint arXiv:2310.13265*.

Xu Zheng, Ziqiao Weng, Yuanhuiyi Lyu, Lutao Jiang, Haiwei Xue, Bin Ren, Danda Paudel, Nicu Sebe, Luc Van Gool, and Xuming Hu. 2025. Retrieval augmented generation and understanding in vision: A survey and new outlook. *arXiv preprint arXiv:2503.18016*.

Junjie Zhou, Zheng Liu, Shitao Xiao, Bo Zhao, and Yongping Xiong. 2024. Vista: Visualized text embedding for universal multi-modal retrieval. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3185–3200.

Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. 2023. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*.

A Appendix

A.1 Answer Generation Evaluation Metrics

A.1.1 Keywords Overlapping Accuracy

Keywords overlapping accuracy is proposed by Chang et al. (2022) that aims to: 1. Detect the presence of key entities. 2. Penalize the use of any incorrect entities. 3. Avoid penalizing semantically relevant but superfluous words. The answer domains D_{qc} are defined for the question categories qc including color, shape, number, Y/N questions, and others. With c as a candidate output, K for correct answer keywords, and qc for question category, the keywords overlapping accuracy is computed by

$$A_{\text{key}}(c, K) = \begin{cases} F_1\left(c \cap D_{qc},\ K \cap D_{qc}\right) \\ \quad \text{if } qc \in \{\text{color, shape, number, Y/N}\} \\ RE\left(c,\ K\right) \\ \quad \text{otherwise,} \end{cases}$$

where RE denotes the recall BARTScore.

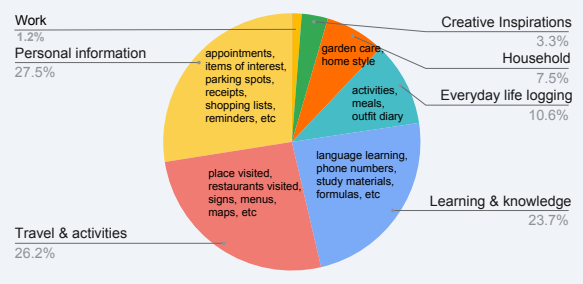


Figure 4: Category distribution of the MemoryQA-test-l dataset, with detailed use cases shown within the pie chart.

A.1.2 Auto Judge

We use an LLM-based (Llama3.3-70B) auto judge to compare the generated answers with the ground truth answers. The prompt for auto judge is shown in Figure 6.

A.1.3 Entailment Score

For the entailment score, we use SCALE (Lattimer et al., 2023), an automatic evaluation method that provides general faithfulness and factuality scores on all text generations. The generated answer is fed into a prompt to detect whether it implies the ground truth answer. Each prompt is then run through Flan-T5 (Chung et al., 2024), a pre-trained sequence-to-sequence LLM, and the resulting logits are used to compute the entailment scores. Specifically, logits are obtained by prompting M with the following: $l = M(\text{“\{premise\} Question: does this imply ‘\{hypothesis\}’? Yes or no?”})$. The entailment probability is then calculated by

$$P_{\text{entail}} = \text{SoftMax}(l[\text{“Yes”}], l[\text{“No”}])[0].$$

A.2 Prompts

A.2.1 Memory Augmentation

Invocation Completion is generated using Llama3.2-90B-vision. Figure 8 shows the complete prompt. We use it to rewrite the user invocation into a complete sentence with an explicit target. Besides, we generate *QA-guided image description* using Llama3.2-90B-vision with the prompt shown in Figure 9.

A.2.2 Datetime matching

We used a temporal matching module leveraging LLM to extract date time information from the raw user question. The prompt is shown in Figure 10.

Dataset	Number of questions			
	Temporal/location constrains			
	time-only	loc.-only	time & loc.	no-constrain
s	633	53	28	612
l	895	335	355	1386
	Aggregation level			
	single-memory		multi-memory	
s	1,326		16	
l	2,797		174	

Table 10: Statistics of the test dataset. The table reports question counts by aggregation level (single- vs. multi-memory) and by temporal/location constraints. Aggregation level indicates how many memories are relevant to a question. Temporal constraints specify a date range of relevant memories (e.g., *"what did I save yesterday"*), while location constraints specify where the memory was created (e.g., *"what was the hotel name I saved in Las Vegas"*).

A.2.3 Answer Generation

For text models, we used the prompt presented in Figure 11 to generate the final answer. Memories are converted to a JSON object with the following fields: memory id, invocation command, visual content (including invocation completion and image description), OCR text, creation date, and address.

For vision models, we make small changes to the prompt above. First, we added image as attachments. To handle multiple retrieved images, we concatenate all images into a single image with a 5 px spacing between images. Second, we added the following line to the instruction part of the prompt:

```
-- Input structure: If there are multiple user
    memories, images from all memories are
    concatenated together. The order of image is
    consistent with the order of user memories.
```

To generate the final answer along with the reference memory id list, we modify the task into

```
Your current task is to answer questions about user
    memory. You need to provide two fields in JSON
    format, {id_list: [""], response: ""}.
```

and added the explanation of id list into the detailed instruction:

```
-- id_list: A list of memory_id of the memories
     used for answering the query.
```

A.3 Dataset

A.3.1 VQA-Mem Dataset

Image and QA Pair Selection We use Llama3.3-70B to remove QA pairs that are not recall questions and answers. The prompt for data filtering is shown in Figure 12 and Figure 13. We only keep images with at least one recall QA pair, ensuring relevance for Memory-QA.

Invocation Commands Generation We use few-shot learning to prompt Llama3.2-90B to generate two different invocation commands for each selected image from the previous step. The prompt used to generate invocation commands are shown in Figure 7.

A.3.2 MemoryQA Dataset

Test set The test set consists of 2,789 images captured using wearable devices. We first prompt VLM to generate synthetic temporal and location metadata for each image. Then, for each image, we craft a set of invocation commands (e.g., *"remember this restaurant"*) and timestamped memory recall questions (e.g., *"where was the restaurant I saved last week"*). The paired image, invocation command, and temporal/location metadata forms a raw memory. To create the memory context for each recall question, we combine the source memory with 10-50 additional memory samples randomly drawn from the memory pool, ensuring all memories are created before the recall timestamp. e then prompt VLM to identify relevant positive memories from this context and generate a final answer based on the selected positive candidates. In the end, human annotator review the selected positive memories and generated answers, making adjustments as needed to ensure high-quality question-answer pairs.

Our test dataset spans diverse categories and use cases. Figure 4 illustrates the distribution of the seven main categories and their associated use cases in the MemoryQA-test-l dataset. Table 10 further summarizes the test set statistics, breaking down questions by temporal and location constraints as well as by aggregation level (i.e., whether a question involves single or multiple memories). Table 5 provides examples that compare the performance of a state-of-the-art baseline (RagVL) with our proposed PENSIEVE , highlighting differences in retrieval accuracy and answer generation.

Train set Our training set follows a similar process, with two key differences. First, the invocation commands and recall questions are generated by an VLM. Second, instead of human validation, we use VLM to filter irrelevant memories from the positive candidates based on the question and generated answer.

A.4 SOTA MM-RAG Implementation

A.4.1 VISTA

Following the methodology in (Zhou et al., 2024), we use the vis-BGE-m3 model as our multi-modal encoder. For each memory entry M_i, the encoder processes the raw image, invocation command, creation datetime, and address to generate an embedding vector. At runtime, we encode the user query using vis-BGE-m3 into a query embedding. Then we compute the dot product between the query and memory embeddings to obtain a similarity score. The top-K memory entries with the highest similarity scores are selected and passed to GPT-4o for answer generation.

A.4.2 RagVL

Our implementation of the RagVL pipeline follows the approach described in (Chen et al., 2024). It comprises two stages: a retriever and a vision-language model (VLM)-based re-ranker.

In the retrieval stage, we use CLIP to encode memory images and retrieve the top 20 most relevant candidates for a given user query. These candidates are then passed to the re-ranking stage using LLaVA-v1.5-13B, which was fine-tuned on the WebQA dataset. Each memory is represented as a JSON-formatted textual passage containing the following fields: invocation command, creation datetime, and address. The re-ranker takes this passage along with the corresponding memory image and the query as input, using the following prompt (adopted from the original paper):

```
Based on the image and its caption, is the image
    relevant to the question? Answer 'Yes' or 'No'.
```

The probability of generating the token 'Yes' is used as the re-ranking score. The final top-K ranked memories are then passed to GPT-4o for answer generation.

A.5 Signal fusion re-ranker

Our signal fusion re-ranker combines multiple relevance scores into a single ranking. We evaluate three fusion strategies:

- Max: We rank memories by its highest score across all retrievers. Ties are broken by comparing the next highest scores.
- Sum: We compute and rank the sum of scores from all signals for each candidate.
- Learned-weight: We obtain the weight of each signal by training a linear Support Vector Classifier (RankSVM) (Joachims, 2002) on a small randomly sampled subset of the training data with squared hinge loss.

As shown in Table 4, the learned-weight approach achieved the best retrieval performance.

Model	WebQA	VQA-Mem	MemoryQA s	MemoryQA l
Baseline (w/o aug., CLIP)				
GPT-4o (vis)	67.6	52.8	53.3	49.1
Llama3.2-90B (vis)	61.0	48.2	52.5	50.5
Llama3.3-70B (txt)	49.4	7.3	34.8	40.0
SOTA MM-RAG systems, w/ GPT-4o				
VISTA	71.0	52.9	58.2	57.2
RagVL	**72.2**	52.5	57.5	58.3
PENSIEVE *: w/ aug., Multi-signal Retriever*				
GPT-4o (vis)	68.7	**55.8**	69.4	66.0
Llama3.2-90B (vis)	64.1	52.0	70.0	65.3
Llama3.3-70B (txt)	59.5	47.5	**71.0**	**66.1**

Table 11: E2E QA results A_{key}. PENSIEVE outperforms the baseline and state-of-the-art solutions on recall questions from VQA-Mem and MemoryQA by a big margin.

A.6 Overall A_{key} Results

E2E QA results A_{key} is shown in Table 11. The score trends in general align with A_{llm} in Table 2, demonstrating the effectiveness of PENSIEVE evaluated under different quality metrics.

A.7 Ambiguous Temporality

To demonstrate the robustness of our system under ambiguous temporal reference, we conduct a focused case study using 10 user queries with vague temporal cues (e.g., *"What was the event we went to a while back?"*, *"What did I save after the meeting?"*, *"Which movie did I watch a while ago?"*). Our findings indicate that in these cases, the datetime matching module often produces an empty search date range and does not trigger the recency signal. As a result, retrieval relies primarily on semantic similarity between the query and the memory content via embeddings.

A.8 Model fine tuning

We fine-tuned the Llama3.1-8B-Instruct model using supervised fine-tuning (SFT) for the answer generation stage. The training was conducted for 4 epochs using the following hyperparameters: learning rate 2×10^{-6}, batch size 1, gradient accumulation steps 4, minimum learning rate ratio 0.1, warm-up steps 200. The best performance was achieved after the first epoch.

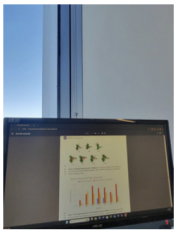
Recall question and ground truth answer	Ground-truth memory	RagVL	Pensieve	Notes
Datetime: 2024-03-25 12:03PM Question: "What software did they use to predict aptamer-protein complex structures?" Answer: HADDOCK	 Memory_id:"0533e0c4" Datetime: 2024-03-15 02:30PM Invocation: remember the details of the graph on the screen	Top candidates: ["0533e0c4", "55695a6a", "dcb39e8e"], Response: "The software used to predict aptamer-protein complex structures is Rosetta."	Top candidates: ["0533e0c4", "1622f64f", "7b151740",] Response: "They used the HADDOCK software to predict aptamer-protein complex structures."	RagVL retrieved the correct memory but GPT-4o failed to produce the right answer. With rich augmentation from OCR, Pensieve retrieved the correct memory and generated the correct answer.
Datetime: 2024-03-07 10:39AM Question: "what did I save last week?" Answer: "You saved information about the mural titled 'One World, Our Children' located in New York, New York."	 Memory_id: "76ce26b0" Datetime: 2024-03-01, 03:15 PM Invocation: remember the mural titled 'One World, Our Children'	Top candidates: ["2b60a783", "c244e87b", "0941b8e4"] Response: "You did not save any memories last week."	Top candidates: ["76ce26b0", "ecaeff22", "18521b9d",] Response: "You saved a memory about the mural titled 'One World, Our Children' by Chinon Maria."	RagVL failed to retrieve memories for time-constrained questions, while Pensieve, with its datetime matching module, successfully parsed the date range and retrieved the correct memory. .
Datetime: 2024-07-02 23:44PM Question: "what did I save the last time I was in New York" Answer: "You saved a reminder to trim the bush in New York."	 Memory_id: "0636de66" Datetime: 2024-06-15, 09:12 AM Invocation: remember to trim this bush	Top candidates: ["0388f3e2", "2b275e99", "0a64aaca"], Response: "You saved memories of the design of the door with the 'La Compagnie' sign and the architectural details of the building entrance at 100 Broadway."	Top candidates: ["3fdaa6b7", "0636de66", "c4c7f56e",] Response: "You saved a memory to remember to trim a bush"	When multiple memories are relevant, Pensieve uses recency signals to rank the correct one higher and answer the question accurately.

Figure 5: Case studies comparing the state-of-the-art RagVL with our proposed PENSIEVE pipeline.

```
You are an evaluator, and you are given a task to evaluate a model predictions with a given question.
     Let's follow the instructions step by step to make a judgement.

1. As the first step, you need to check whether the prediction was really answering the question.


2. If the model prediction does provide a meaningful answer, judge whether the model Prediction matches
     the ground truth answer by reasoning according to the following steps:

2.1: Always assume the ground truth is correct.


2.2: Pay attention to theses special cases:


    a. If the ground truth answer contains numbers,  the value of "accuracy" is true only if numbers in
     ground truth and numbers in model predictions match very well; in case of math questions, "accuracy

     " is true only if the numbers in model predictions EXACTLY matches the numbers in ground truth;


    b. If the ground truth answer contains time, and/or time range, "accuracy" is "true" only if if
     times and time ranges in ground truth and model predictions match very well.

    c. If the ground truth answer contains a set of objects, "accuracy" is "true" if the model
     prediction covers most of the objects in the ground truth; however, "accuracy" if "false" if the

     model prediction has a lot of objects that are not in the ground truth.


    d. If the ground truth is something similar to "I don't know", "accuracy" is "true" only if the
     model prediction also implies the similar thing.

2.3: Even if the prediction statement is reasonable, if it conflicts with or does not match the ground
     truth, "accuracy" should be "false".

2.4. "Accuracy" is true if the ground truth information is covered by the prediction. The prediction is
     allowed to provide more information but should not be against the ground truth. If it is hard to

     decide whether the prediction matches ground truth, "accuracy" should be "false".


Think step by step following the instructions above, and then make a judgment. Respond with only a
     single JSON blob with an "explanation" field that has your short(less than 100 word) reasoning

     steps and an "accuracy" field which is "true" or "false".


Question: {{question}}


Ground truth: {{answer}}


Prediction: {{prediction}}
```

Figure 6: Prompt for Llama3.3-70B to auto-judge the response, where {{prediction}} is the response from answer generator.

```
You are a helpful assistant that can generate evaluation data for a memory save and retrieval stack. The
      stack helps users to remember what they saw in the image and allows them to retrieve the memory

     and ask questions about it. Given an image, your task is to generate these items:

(1) "invocation1": a user query to create the main memory of what the user sees.

(2) "invocation2": a different user query to create another memory of what the user sees.



For example,

[image is about a LG refrigerator with a price tag]

Response:

{"invocation1": "remember the fridge",

"invocation2": "remember the price"}



Now look at the image and generate the items.
```

Figure 7: Invocation commands generation prompt.

```
## Task Description

You are a skilled assistant capable of completing invocation sentences based on an image.



### Key Definitions

* **Invocation Sentence**: A concise transcription associated with an image object, capturing key
    information for later recall.
* **Invocation Completion**: A completed invocation sentence with additional details about the object,
    such as attributes, actions, or context.



Example:

Invocation Sentence: "remember the restaurant"

Invocation Completion: "remember the Korean restaurant named 'Kochi' in NYC"



## Input

You have been provided with an invocation sentence for the image:

{{invocation}}



## Output Requirements

Please analyze the image and generate an invocation completion for the invocation sentence.
```

Figure 8: Prompt for VLM to rewrite the invocation into a complete sentence with explicit target.

```
## Task Description

You are a skilled assistant capable of generating recall questions and answers based on an image, as
     well as creating a detailed image description that addresses all the recall questions.



### Key Definitions

* **Recall Question**: A user query about an image from their past, aiming to retrieve relevant
     information among all images at a later time.

  Examples:

  - What is the name of the Korean restaurant?

  - Where did I park my car today?

  - Where is the bedroom key?

  - When is the milk expiration date?

  - What vegetables do I have in my fridge?

  Non-examples:

  - What is the girl doing in the image?

  - Why are there stickers on the oranges?

  - What time is it?

  - Where is the bench located in the image?

  - What object is on the right side of the image?

* **Recall Answer**: A precise response to a recall question, enabling information recall without visual
      reference.



## Output Requirements

Given an image, provide the following items in JSON format:

  - 'recall_question': A list of potential recall questions a user might ask about the image.

  - 'recall_answer': A list of corresponding recall answers for each recall question.

  - 'image_description': A comprehensive image description with additional details that address all the
     recall questions above.



Please analyze the provided image and generate the required items.
```

Figure 9: QA guided image description prompt

```
Given a user question recalling a saved memory and its timestamp, extract the search_start_date and
     search_end_date of the user question. Also, predict whether the user wants to search for the most

     recent memory or not.

- search_start_date: The start date of the search range in the database, formatted as "YYYY-MM-DD" (e.g
     ., "2024-08-25").
- search_end_date: The end date of the search range, also formatted as "YYYY-MM-DD" (e.g., "2024-08-25")
     .
- search_recent: A boolean value indicating whether the user wants to search for the most recent memory.

- If no time information is provided in the question, set search_start_date and search_end_date to empty
      strings ("").



For example:

question: "where did I park yesterday"

recall_time: "2024-05-06 Tuesday"

output:

{"search_start_date": "2024-05-05", "search_end_date": "2024-05-05", "search_recent": false}



question: "which book did I saved last time"

recall_time: "2024-08-26 Monday"

output:

{"search_start_date": "", "search_end_date": "", "search_recent": true}



Here is the user question and recall time:

question: {{question}}

recall_time: {{recall_time}}

Now generate the output in JSON format without any other text.
```

Figure 10: Prompt for Llama3.3-70B to parse the search date time and recency signal from the raw user query.

```
### Instruction:

You are an assistant. Your current task is to answer questions about user memory.

Here are detailed instructions:

  -- Input structure: When given a user memory, it will contain: memory_id, created_datetime,
     description, visual_content, ocr_text.

  -- Input structure: visual_content is a description of the image attached to the user memory, and
     ocr_text is the text extracted from the image. Both are optional and might not be available.

  -- Response format: Be terse and to the point, don't mention your reasoning, and answer in a single
     sentence.

Now look at all the content in all given user memories, and provide "response".

### Input:

Current date time is: {current_date_time}

Candidate memories: {memory_candidates}

Current turn:

- User: {user_query}

### Response:
```

Figure 11: Answer generator prompt where {{memory candidates}} are memories in a JSON format.

```
You are a helpful assistant that can identify real recall questions for a memory save and retrieval
     stack.



Group Definition: Recall Question

Description: A query posed by a user regarding an image from their past, with the intention of
     retrieving associated useful information at a later time.
The following sentences belong to the group "Recall Question":

- What is the brand of the milk powder?

- What is the name of the korean restaurant?

- Where did I park my car today?

- Where is the bathroom key?

- When the monthly fees are due?

- What kind of plants do I have in my garden?

- What is my license plate number?

The following sentences do not belong to the group "Recall Question":

- What is the girl doing?

- What color is the stop light?

- Why are there stickers on the oranges?

- What time is it?

- Where is the man standing?

- What is the word that starts with 'W'?

- What object is in focus?



Does the following sentence belong to the group "Recall Question"? Answer only with "True" or "False".
```

Figure 12: Recall question filtering prompt.

```
You are a helpful assistant that can identify good recall answers for a memory save and retrieval stack.



Group Definition: Recall Answer

Description: A precise response to a user's query about an image from their past, enabling recall of
     associated information without visual reference.
The following sentences belong to the group "Recall Answer":

- Question: Where are the blue container bins? Answer: under the dinning table

- Question: What phone number is on the sign? Answer: 604-909-7275

- Question: What type of condiment is on the top shelf? Answer: mayonnaise

- Question: Where is the bus parked by? Answer: main st

- Question: What brand is the bike? Answer: yamaha

The following sentences do not belong to the group "Recall Answer":

- Question: Where are the blue container bins? Answer: left

- Question: What phone number is on the sign? Answer: 0

- Question: What type of condiment is on the top shelf? Answer: condiment

- Question: Where is the bus parked by? Answer: m

- Question: What brand is the bike? Answer: no brand



Does the following sentence belong to the group "Recall Answer"? Answer only with "True" or "False".
```

Figure 13: Recall answer filtering prompt.