nature portfolio

Peer Review File

Ubiquitous Memory Augmentation via Mobile Multimodal Embedding System

Corresponding Author: Mr Dongqi Cai

This file contains all reviewer reports in order by version, followed by all author rebuttals in order by version.

Version 0:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
The paper proposes the Reminisce framework, an efficient on-device multimodal embedding system. It is a coarse-grained embedding approach based on existing techniques. The model first generates coarse-grained embeddings used to filter out candidates during retrieval queries. The most promising candidates are further refined at query time to achieve accurate retrieval. This strategy aims to significantly reduce the hardware requirements of the model, enabling deployment on mobile devices.
Main Comments:
The introduction extensively explains the effects of the proposed approach, but it does not mention the techniques used. How do the authors reduce the computational cost of the search and embedding procedure? A brief technical introduction to the employed techniques is necessary.
The authors state: “Since ORIN’s GPU does not support INT4 computation, we load the raw model with FP32 precision.” I understand that Jetson devices do not support INT4, but ORIN supports INT8, and the Nano can efficiently handle FP16 quantization levels. Using FP32 seems highly suboptimal. A similar observation applies to the Raspberry Pi. Additionally, Jetson devices offer various operating modes, ranging from energy-efficient setups to maximum performance configurations. What was the specific setting used? The same concern applies to many other experiments. Given that deployment results play a major role in the proposed paper, the lack of a rigorous analysis is a significant issue.
The proposed methodology appears to be a combination of existing techniques. The proposed pipeline heuristically balances the computational cost and performance of the method. However, similar results could be achieved in various ways. I do not see a clear advantage over other possible methodologies. The authors should more clearly explain the novelty of their approach compared to existing methods.
Punctual Observations
• In Figure 5, the authors present results for the Jetson TX2, while in Figure 2 it is not included. Why?
• Table 2 is extremely terse. A different way to visualize these data could help readers better understand the benefits of the proposed approach. Additionally, the observed out-of-memory issue could likely be avoided by using FP16 or INT8 quantization.
• The section “Significance of Key Designs” is introduced before sufficiently explaining the key designs.
• Figure 7 presents the inference time without specifying the target devices or the quantization method used. The search time for most thresholds is still far from real-time, reaching up to 10 seconds.
• The same concern applies to Figure 8. What device was used? What was the measurement setup?
• In the subsection “Case Study: Twitter Meme Retrieval,” the analysis is too shallow.
Minor Issues:
• Unclear sentence: "The generated embeddings can be easily retrieved to help mobile users remember and recall information when needed." The meaning should be clarified.
• Typo: "Based on the trace statistics in §, typical" – The reference format needs correction.
• Typo: "This method outperforms fixed early-exit baselines, as will be shown in §." – The section reference should be properly formatted.

(Remarks on code availability)

Reviewer #2

(Remarks to the Author)
General Comments:

The manuscript presents a valid contribution to the field of multimodal embedding, specifically focusing on efficient on-device computation for resource-constrained devices, including mobile. The proposed system, Reminisce, introduces a novel approach that enhances retrieval efficiency while maintaining computational feasibility. The emphasis on hardware-friendly design is particularly valuable, ensuring practical deployment of such models on edge devices and closer results to real-world scenarios. Additionally, the study demonstrates transparency regarding resource consumption, including memory and disk usage.

What are the noteworthy results?

The work achieves significant gains in energy efficiency and throughput, making it well-suited for real-world deployment on resource-constrained edge devices. The evaluation on physical (real) hardware further strengthens its practical relevance and reliability.

Will the work be of significance to the field and related fields? How does it compare to the established literature? If the work is not original, please provide relevant references.

Yes, this work is significant to the field.

It enhances multimodal embedding with a strong focus on hardware efficiency for edge (and mobile) devices. By optimising retrieval in constrained environments, it extends real-world applicability beyond prior research.

Does the work support the conclusions and claims, or is additional evidence needed?

Yes. However, a clearer explanation of the caching invalidation strategy is necessary, as multiple caching mechanisms are involved (as pointed out in Figure 11).

Are there any flaws in the data analysis, interpretation, and conclusions? Do these prohibit publication or require revision?

No.

Is the methodology sound? Does the work meet the expected standards in your field?

The methodology is appropriate and meets field standards.

The approach to on-device computation is particularly strong, as it reduces energy consumption while maintaining retrieval performance, and the motivation (privacy concern when using Cloud solutions) is solid.

Is there enough detail provided in the methods for the work to be reproduced?

Most aspects are well-described, but the caching invalidation mechanism should be clarified for readers.

Final Recommendation:

The study is well-executed and makes a valuable contribution to the field. Addressing minor issues related to caching, terminology, and figure corrections will further enhance the clarity of the manuscript.

Figure corrections are: Figure 6 ("Ous" - > "Ours"). Figure 3b requires clarification in the legend regarding the meaning of the black and white bars colours.

Minor terminology inconsistencies: the misuse of "backend/frontend" where "background/foreground" would be more appropriate. Similarly, "backward" is incorrectly used instead of "background" in the appendix.

Additionally, some redundant sentences in the appendix, such as "This causes the application to be easily killed by the OS due to memory pressure," should be removed to improve readability.

With these refinements, I strongly support its publication.

(Remarks on code availability)
My assessment of the code is that it is a usable resource for the community, with clear details on the installation, and use of the available scripts. It also cares for important (Open-Source)OS details, such as the clear deinition of its OS license.

Version 1:

Reviewer comments:

Reviewer #1

(Remarks to the Author)
The authors successfully replied to all my questions and concerns. I have no further requests.

(Remarks on code availability)
The authors successfully replied to all my questions and concerns. I have no further requests.

Reviewer #2

(Remarks to the Author)
The revised manuscript presents a meaningful and well-executed contribution to the field of multimodal embedding, with a particular emphasis on efficient on-device computation for resource-constrained platforms, including mobile devices.

The authors have addressed the previously raised concerns regarding caching mechanisms, terminology, and figure clarity. The cache invalidation strategy has been clarified, and the necessary corrections to figures and textual inconsistencies have been made. These refinements have significantly improved the clarity and precision of the manuscript.

With these improvements in place, I strongly support the publication of this work. The study is both technically sound and practically relevant, offering valuable insights for deploying efficient multimodal retrieval systems on non-simulated edge devices.

(Remarks on code availability)
I maintain my previous assessment of the code: which is that it is a usable resource for the community, with clear details on the installation and use of the available scripts. It also cares for important (Open-Source)OS details, such as the clear definition of its OS license.

GENERAL RESPONSES

We thank the reviewers for their insightful comments and helpful suggestions. The original comments are attached at the end of this document. We have carefully revised our manuscript according to these comments and responded to each point individually. The modifications made to the manuscript are highlighted in blue text or enclosed within blue boxes after each response for clarity. Additionally, due to space constraints in the main text, several supplementary experiments and extended discussions have been included in the supplementary materials. We hope that our revisions and responses adequately address all concerns.

A summary of major revisions are provided below:

- **Further Design and Result Clarification**
  - **Core Techniques:** We expanded the introduction with a concise overview of our core techniques to clarify how embedding costs are reduced (Reviewer #1–Comment 1: Addressed in lines 83–109).
  - **Experimental Settings:** We clarified the quantization and deployment settings, specifying default experimental conditions and detailing performance modes (Reviewer #1–Comment 2: Addressed in Figure 4, Table 2, lines 307–322, Supplementary Table 3, etc.).
  - **System Advantages:** We revised the results and methods sections to emphasize our system's advantages compared to prior literature (Reviewer #1–Comment 3: Addressed in Figure 3, §2.3 (lines 197–249), lines 567–589, etc.).
  - **Invalidation Strategy:** We added a description and illustrative workflow of our cache invalidation strategy to the methods section (Reviewer #2–Comment 4: Addressed in Figure 9, lines 776–787).
- **Enhanced Evaluation**
  - **Integration with Quantization:** All evaluation experiments have been updated under the appropriate quantization settings, demonstrating that our method, being orthogonal to and highly compatible with the quantization scheme, consistently achieves superior performance compared to previous methods.(Reviewer #1–Comment 2: Addressed in §2.5).
  - **Optimized Query Efficiency:** Query efficiency was improved by loading INT8 models entirely into RAM. This accelerated the fine-grained query phase and reduced query latency to under 1.5s, only 300ms higher than naive retrieval system, while achieving up to 31× higher embedding throughput. (Reviewer #1–Comment 7: Addressed in §2.5.3).
  - **Further Use Case Study Analysis:** The use case study has been expanded to include comparisons with additional baselines, accompanied by a more in-depth discussion (Reviewer #1–Comment 9: Addressed in §2.5.5).
- **Compliance with Editorial Requests**
  - **Formats**: We reformatted the manuscript to adhere to *Nature Communications* formatting instructions, including author contributions, competing interests, and supplementary information etc.
  - **Data Availability:** Descriptions of all four previously published datasets and our self-collected dataset have been provided. Anonymization and consent information for real user data have also been included.
  - **Author Information:** Author information has been updated. Two new authors have been added to assist with revisions, and detailed explanations along with consent letters have been provided. All corresponding authors now have ORCID IDs.
  - **Others**: The checklist has been updated accordingly. Typos have been corrected. The manuscript has been proofread and polished.

RESPONSES TO REVIEWER #1

- **Comment 1:** The introduction extensively explains the effects of the proposed approach, but it does not mention the techniques used. How do the authors reduce the computational cost of the search and embedding procedure? A brief technical introduction to the employed techniques is necessary.
- **Response 1:** We thank the reviewer for pointing out the need to clarify the technical contributions of our approach. In response, we have revised the introduction section (lines 83–109) to include a concise and focused summary of the key techniques employed to reduce the computational cost of the search and embedding procedures. We hope this addition addresses the reviewer's concern and improves the clarity of our methodological contribution.

Revised manuscript line 83–109

`Reminisce` is the first-of-its-kind efficient on-device multimodal embedding system. Its key idea is *coarse-grained embedding*, built upon the early-exiting technique. It draws inspiration from the top-down predictions of cognitive brain [17]. Embeddings from early-exited MEMs serve as coarse-grained representations to filter likely candidates during retrieval. These candidates are then refined by the remaining layers at query time for final selection. While early exiting avoids full model execution during memorization, three key system challenges remain on mobile devices (§4.1): low parallelism, limited exiting benefits, and performance degradation. To further promote the practical deployment of `Reminisce`, we propose three novel software-hardware co-designs: (1) *Data-aware pre-exit predictor* (§4.2) is a unified, lightweight early-exit predictor model applicable across all modalities. It facilitates efficient batching and pipeline execution, significantly improving encoding throughput; (2) *Progressive LoRA healing* (§4.3) retrofits low-rank adaptation (LoRA) [17], a popular parameter-efficient fine-tuning method, to ensure high retrieval performance with earlier exits by progressively increasing shared bottom layers. This enables intermediate results to be cached and reused; (3) *Speculative fine-grained retrieval* (§4.4). Query embeddings from different exits are used for speculative filtering, with top candidates from each granularity undergoing a second matching round for accurate final retrieval.

- **Comment 2:** The authors state: "Since ORIN's GPU does not support INT4 computation, we load the raw model with FP32 precision." I understand that Jetson devices do not support INT4, but ORIN supports INT8, and the Nano can efficiently handle FP16 quantization levels. Using FP32 seems highly suboptimal. A similar observation applies to the Raspberry Pi.
  Additionally, Jetson devices offer various operating modes, ranging from energy-efficient setups to maximum performance configurations. What was the specific setting used? The same concern applies to many other experiments. Given that deployment results play a major role in the proposed paper, the lack of a rigorous analysis is a significant issue.
- **Response 2:** We thank the reviewer for the insightful comment regarding the use of lower-precision quantization and the operating modes of the Jetson devices. We have changed the quantization settings and clarified hardware configurations in the revised manuscript to address these concerns (Table 2, line 307–322). Please refer to the following Responses 2a (to lower-precision quantization) and 2b (to operating modes) for more details.

Device	Specification	Model Precision
Jetson Orin [23]	1024-core NVIDIA Ampere GPU. Default operation mode: MAXQ.	INT8
Jetson TX2 [36]	256-core NVIDIA Pascal GPU. Default operation mode: MAXN.	INT8
Raspberry Pi 4B [37]	Broadcom BCM2711B0 quad-core A72 64-bit @ 1.5GHz CPU.	INT8
Redmi Turbo 3 [38]	Qualcomm Snapdragon 8 Gen 3 CPU. Default operation mode: Balance.	INT4

Table 2. Experimental devices used in experiments.

Revised manuscript line 307–322

Hardware and Quantization As summarized in Table 2, we evaluate `Reminisce` on the NVIDIA ORIN (ORIN) [24], Jetson TX2 (TX2) [37], Raspberry Pi 4B (RPI4B) [38], and a flagship smartphone with Qualcomm Snapdragon 8Gen3 (8GEN3) [39]. The default operating mode for ORIN is MAXQ, which is the most cost-effective mode with four large cores disabled. For the Jetson TX2, we select the MAXN mode, the most powerful mode available, to fully utilize GPU computing power. To reduce memory consumption, we quantize the model to INT4 precision for the 8GEN3 smartphone and INT8 precision for ORIN, TX2, and RPI4B. Please refer to Supplementary §G for more implementation details about executing mode specifications and quantization. `Reminisce` runs on the GPU for the ORIN and TX2 boards. For the RPI4B and the 8GEN3 smartphone, `Reminisce` runs on the CPU due to the lack of CUDA support.

- **Response 2a:** In the revised manuscript, we have quantized the model to INT8 across all three development boards. Our approach remains clear performance improvements over the corresponding baselines under INT8 precision, demonstrating that our method, being orthogonal to and highly compatible with the quantization scheme (Supplementary Table 2). Due to space constraints, this table has been moved to the Supplementary, with its illustrative content Figure 4 retained in the main text. All experimental results have been revised accordingly under the new quantization setting (Figure 4–8, 12, 14). The figures captions are updated accordingly to state the experiment setting clearly. Please refer to §2.5 of the revised manuscript for detailed analysis.

Dataset	COCO					FLICKR					CLOTHO					HARSMART				
Throughput (Contents / s)	Relative Accuracy	ORIN (INT8)	TX2 (INT8)	RPI4B (INT8)	8GEN3 (INT4)	Relative Accuracy	ORIN (INT8)	TX2 (INT8)	RPI4B (INT8)	8GEN3 (INT4)	Relative Accuracy	ORIN (INT8)	TX2 (INT8)	RPI4B (INT8)	8GEN3 (INT4)	Relative Accuracy	ORIN (INT8)	TX2 (INT8)	RPI4B (INT8)	8GEN3 (INT4)
MEM	100%	1.98	0.34	0.05	0.05	100%	1.98	0.34	0.05	0.05	100%	5.34	3.15	0.30	0.27	100%	34.15	4.30	1.03	0.81
MEM (Batched)		17.45	1.36	0.07	0.10		17.45	1.36	0.07	0.10		58.41	5.28	0.38	0.32		121.41	7.86	1.28	0.96
BranchyNet	71%	2.96	0.50	0.08	0.07	93%	2.27	0.38	0.06	0.06	81%	31.12	7.96	0.76	0.69	57%	108.98	13.72	3.29	2.58
Fluid Batch		26.08	2.03	0.11	0.16		20.01	1.55	0.08	0.12		100.71	13.34	0.96	0.80		387.49	25.07	4.08	3.05
Ours	95%	62.31	3.31	0.15	0.31	95%	44.33	2.32	0.10	0.22	98%	240.91	25.64	0.99	0.84	95%	787.46	69.51	6.88	5.15
MFM (w/o layerwise)	100%	36.49	1.70	0.07	0.16	100%	36.49	1.70	0.07	0.16	100%	125.52	5.85	0.40	0.34	100%	191.24	8.91	1.33	1.02
BranchyNet (w/o layerwise)	71%	54.53	2.54	0.11	0.25	93%	41.83	1.95	0.09	0.19	81%	317.11	14.77	1.00	0.85	57%	610.33	28.43	4.23	3.25
Ours (w/o layerwise)	95%	72.16	3.36	0.15	0.33	95%	50.44	2.35	0.10	0.23	98%	317.11	14.77	1.00	0.85	95%	1024.49	47.72	7.10	5.45

Supplementary Table 3: Throughput vs. relative retrieval accuracy.

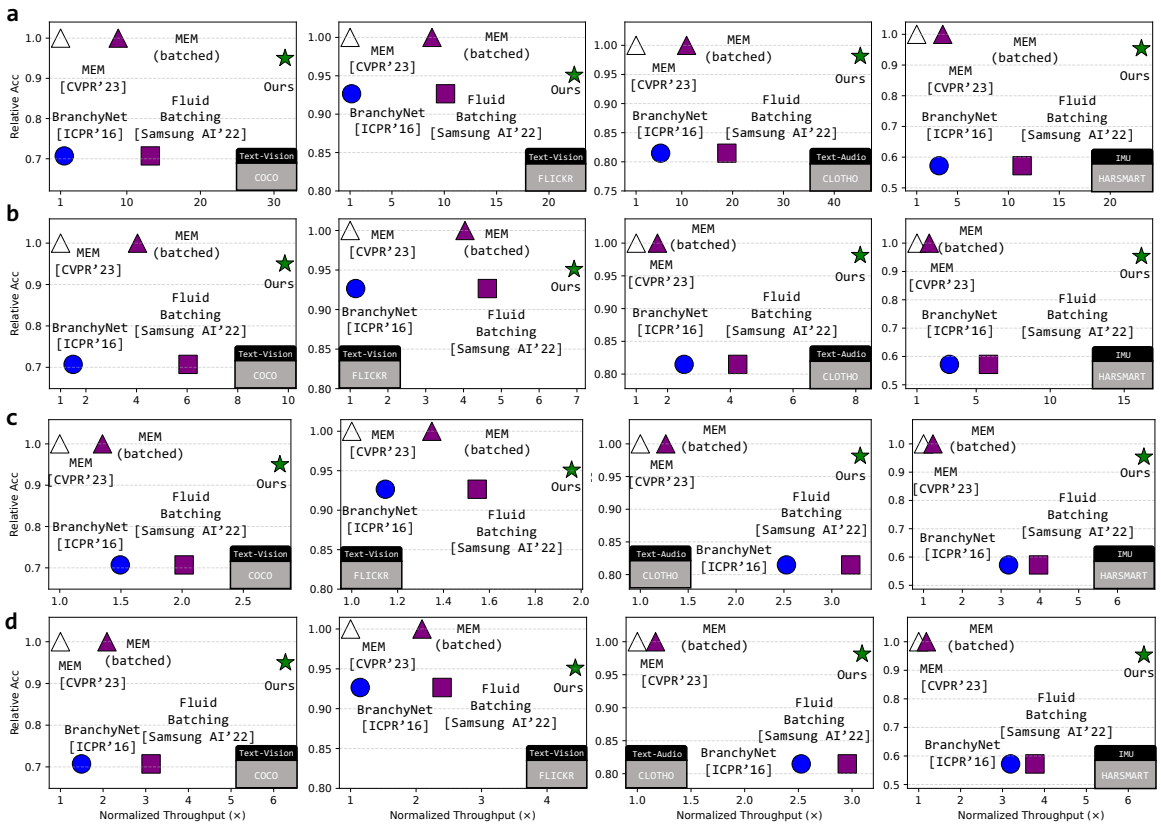


Figure 4: Illustration of throughput versus accuracy for different methods on different devices. (a) Jetson Orin (INT8). (b) Jetson TX2 (INT8). (c) Raspberry Pi 4B (INT8). (d) 8Gen3 Smartphone (INT4). For fairness, only layerwise baselines are included.

- **Response 2b:** As for executing mode, we specify the mode as MAXQ for Jetson ORIN to save energy. For Jetson TX2, we specify the mode as MAXN to maximum the throughput. All of those choices have been clarified in the revised manuscript (line 311–315). The hardware specifications of different executing modes are listed in Supplementary Table 1/2 for a detailed description. Besides, we include additional results in terms of how different modes affect end-to-end inference latency (Supplementary Figure 7, supplementary line 1137–1147).

Revised manuscript line 311–315

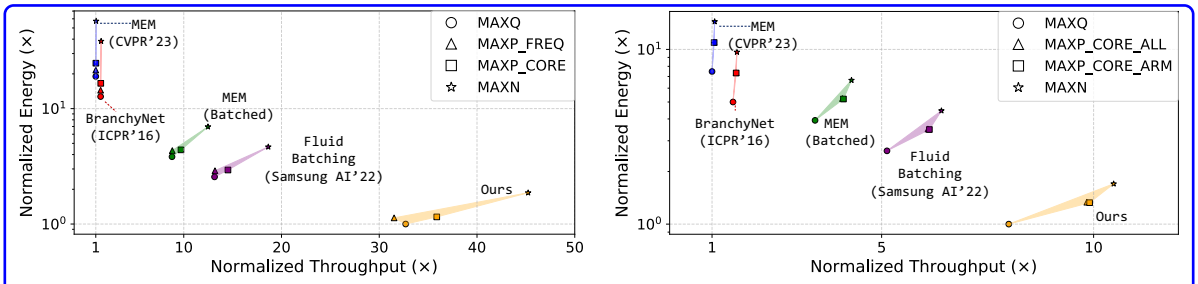
The default operation mode of ORIN is MAXQ, which is the most cost-effective mode, with 4 big cores off. While for Jetson TX2, we select the MAXN mode, the most powerful mode by default, to fully utilize the GPU's computing power.

Supplementary Table 1. Jetson Orin Power Modes Overview.

ID	Name	CPU Online	Max CPU Freq	Max GPU Freq
0	MAXN	0–7 all	729.6 MHz–9.22 GHz	0–9.22 GHz
1	MAXQ	0–3	729.6–1190 MHz	0–612 MHz
2	MAXP_FREQ	0–3	729.6–1420 MHz	0–612 MHz
3	MAXP_CORE	0–7 all	729.6–1497.6 MHz	0–408 MHz

Supplementary Table 2. Jetson TX2 Power Modes Overview.

ID	Name	CPU Online	Max CPU Freq	Max GPU Freq
0	MAXN	1–5 all	2.14 GHz	2.14 GHz
1	MAXQ	3–5 on	1200 MHz	0–850 MHz
2	MAXP_CORE_ALL	1–5 all	1400 MHz	0–1120 MHz
3	MAXP_CORE_ARM	3–5 on	2000 MHz	0–1120 MHz



(a) Jetson Orin (INT8). (b) Jetson TX2 (INT8).

Supplementary Figure 7: System performance under different operating modes. Dataset: COCO.

Revised supplementary line 1137–1147

Furthermore, as shown in Figure 7, more aggressive operating modes generally lead to higher system throughput. For example, on ORIN, using MAXN mode provides a 1.4× throughput improvement compared to MAXQ mode, albeit at a 6.3× increase in power consumption. Therefore, selecting an appropriate operating mode based on the specific device characteristics and requirements is crucial for optimizing the performance-energy tradeoff. Fortunately, Reminisce consistently demonstrates superior efficiency across different execution modes, enabling a more flexible trade-off between throughput and energy consumption.

- **Comment 3:** The proposed methodology appears to be a combination of existing techniques. The proposed pipeline heuristically balances the computational cost and performance of the method. However, similar results could be achieved in various ways. I do not see a clear advantage over other possible methodologies. The authors should more clearly explain the novelty of their approach compared to existing methods.
- **Response 3:** Thank you for the thoughtful comment. Our work introduces the first end-to-end, on-device multimodal embedding system, incorporating a series of optimizations specifically designed for resource-constrained mobile devices to achieve superior efficiency and effectiveness. We have clarified the methodological novelty and its advantages over existing approaches as below:
  - **Coarse-grained embedding:** The core design of Reminisce lies in its coarse-grained embedding strategy, built atop early-exiting (Figure 3a, lines 198–210). Coarse-grained embeddings, generated by early-exited MEMs, are used to filter likely candidates during retrieval queries. These candidates are then refined by the remaining layers of the exited MEMs at query time for accurate retrieval. This design is inspired by the top-down predictions of cognitive brains (line 87). The brain initiates top-down predictions based on partially processed input, and refines them with detailed information. To the best of our knowledge, this is the first efficient

multimodal embedding system designed for mobile devices.
Unlike prior methods (e.g., pruning, quantization), our approach preserves the original model structure, avoids the need for advanced implementation which is often unsupported on mobile (lines 568–573), enables to reuses intermediate activations to significantly reduce redundant computation during embedding and query phases (lines 574–584), and offers a border trade-off space for performance and accuracy (lines 584–589).

- **Mobile-specific optimizations:** We introduce three hardware-software co-designed optimizations that significantly enhance `Reminisce`'s performance, making it practical and efficient for mobile deployment. These optimizations address limitations overlooked by existing early-exit methods, which are primarily tailored for single-modality tasks on desktop or cloud platforms. Previous early-exiting approaches neither account for mobile-specific resource constraints nor effectively incorporate coarse-grained embeddings for multimodal data (Supplementary lines 898–932). (1) *Data-aware pre-exit prediction* (Figure 3b) enables efficient batching and pipelined execution, unlike traditional early-exit methods that determine exits after each branch computation and are incompatible with batching (lines 211–224). (2) *Progressive LoRA healing* (Figure 3c) adapts low-rank adaptation (LoRA) to progressively share adapted parameters, allowing reuse of intermediate LoRA activations across exits while minimizing performance degradation (lines 226–239). (3) *Speculative fine-grained retrieval* (Figure 3d) leverages query embeddings from different exits for speculative filtering, addressing imbalanced retrieval performance when matched with coarse-grained multimodal embeddings (lines 240–249).
- **Empirical evaluation:** Our extensive experiments in §2.5 demonstrate that, with these designs, `Reminisce` outperforms exisiting methods while ensuring accurate retrieval. We evaluate `Reminisce` on multiple mobile devices, achieving an average 12.4× improvement in throughput compared to the original MEM (lines 334–350). Even compared to advanced baselines like `Fluid Batching`, `Reminisce` achieves a 2.4× speedup in throughput (lines 351–360). Each design contributes to the overall performance improvement (lines 362–388). We further conduct a case study using recent Twitter data and a user study based on mobile application traces collected from eight users over one week, demonstrating the practicality of `Reminisce` in real-world scenarios (lines 437–489).

Revised manuscript line 198–208: Description of coarse-grained embedding

As shown in Figure 3a, the core design of `Reminisce` is the coarse-grained embedding, built upon the early-exit mechanism. This approach offloads the computation of the full embedding to the less frequent, intent-specific query phase. Specifically, embeddings generated by early-exited MEMs serve as coarse-grained embeddings to filter the most likely candidates during retrieval queries. These candidates are further refined by the remaining layers of the exited MEMs at query time to ensure accurate retrieval. We are the first to propose and prototype a mobile-friendly early-exit system for efficient multimodal embedding.

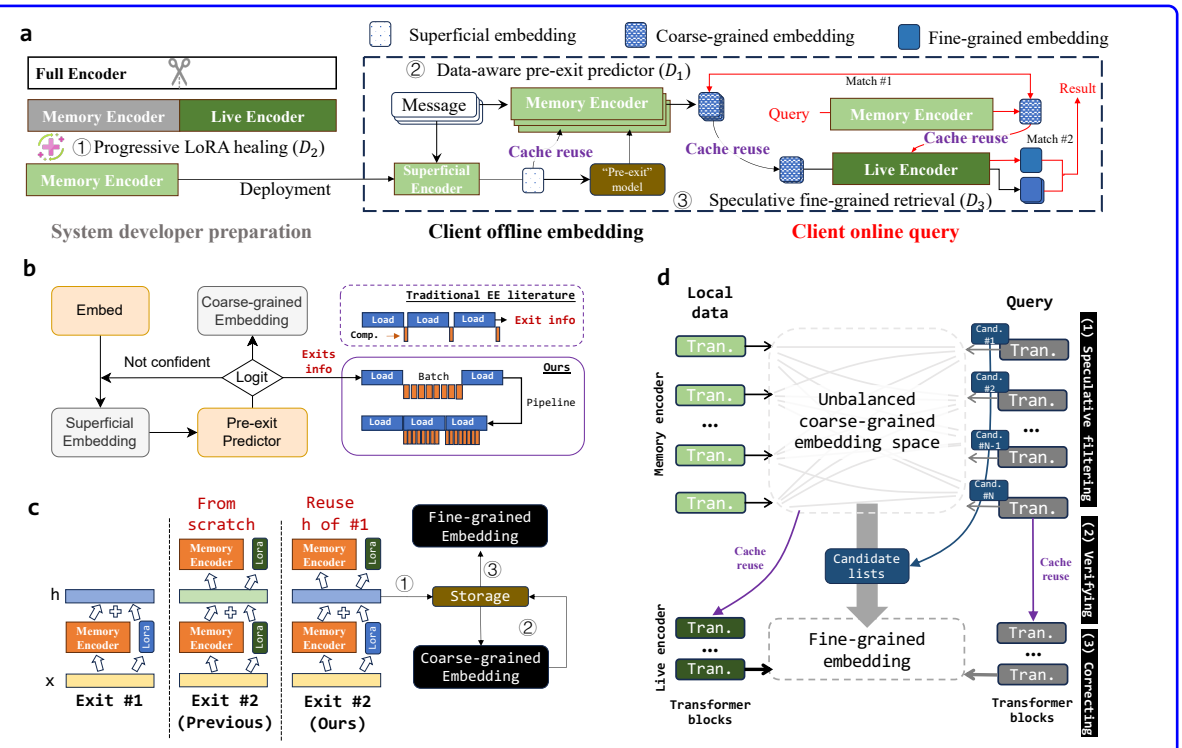


Figure 3. Illustration of the proposed Reminisce. (a) Concrete workflow of Reminisce with system Designs$_{1,2,3}$. (b) Illustration of Design 1: Data-aware pre-exit and its advantage over traditional early exiting. (c) Illustration of Design 2: Comparison between our progressive LoRA and previous methods. (d) Illustration of Design 3: Coarse-grained embeddings are speculatively filtered. The highest-ranking embedding candidates are refined to fine-grained embeddings for final retrieval.

Revised manuscript line 567–589: Advantage of Our Coarse-grained Embedding

We choose early exit as the backbone of Reminisce because it aligns with our design principles: (1) Early exit is mobile hardware-friendly: it requires no sparsification kernel compilation and integrates easily into existing multimodal embedding applications. Most mobile devices do not fully support advanced sparsification or quantization optimizations, providing little to no benefit during inference [45–49]. (2) Early exit preserves the raw structure of MEMs, maintaining their generalization capacity while bypassing only downstream alignment. Additionally, early exit is caching-friendly, as the top layers share the same bottom weights with the exited layers, allowing intermediate activations to be reused and reducing duplicated computations. Other techniques like pruning and quantization cannot fully leverage the intermediate computation of coarse-grained embeddings. This reduction is crucial for Reminisce, as it eliminates redundant forward passes, accelerating both embedding and query phases, which we discuss in §4.5. (3) Compared to quantization, early exit offers a broader trade-off space. As shown in our experiments (Supplementary Figure 4a), easy inputs require only one layer (just 3% of total computation) to achieve accurate results. Such a large reduction in cost is not possible with quantization.

Revised manuscript line 211–249: Mobile-Specific Optimizations

Data-aware pre-exit prediction (§4.2) Traditional early-exit methods determine exits at the end of each branch computation, causing inconsistent workloads and memory fragmentation [26], and existing predictive models for CNNs cannot effectively scale to multimodal embedding models due to their convolution-specific design [27, 28]. Our observation is that different data inherently carry varying amounts of information (Supplementary Figure 4a), and intermediate multimodal embeddings provide effective cues for determining optimal exit points (Supplementary Figure 4b). Based on this unique observation, we propose a unified, lightweight early-exit predictor that leverages these intermediate embeddings to preemptively determine the exit layer, enabling batch scheduling for improved parallelism and amortizing loading times (Figure 3b).

Progressive LoRA Healing (§4.3) Previous early-exit healing approaches [29] utilize LoRA [18] to fine-tune NLP models for earlier exits. However, these methods fine-tune separate LoRA modules for each exit, preventing the reuse of intermediate results and thereby negating early-exit benefits on mobile devices. As illustrated in Figure 3c, we propose sharing previously tuned parameters, significantly reducing the number of layers required per token and enabling reuse of intermediate activations. Based on our observation that sharing LoRA weights at top layers is more effective (Supplementary Figure 4), we propose a progressive LoRA healing method that incrementally increases tuning depth (number of shared layers) at later exits to minimize performance degradation from shared LoRA weights.

Speculative Fine-grained Retrieval (§4.4) Using a full-capacity encoder to generate query embeddings leads to unbalanced retrieval performance when matched with coarse-grained embeddings, resulting in poor top-1 retrieval accuracy (Supplementary Figure 6). To address this issue, we introduce a speculative fine-grained retrieval mechanism (shown in Figure 3d) to balance the retrieval process. It first performs speculative filtering using query embeddings at all granularities and then refines the selection through a second, fine-grained matching stage.

- **Comment 4:** In Figure 5, the authors present results for the Jetson TX2, while in Figure 2 it is not included. Why?
- **Response 4:** We thank the reviewer for pointing out this inconsistency. In the initial submission, we presented partial results in figures and tables to save space. In the revised manuscript, we have added new illustrations (see Figure 4 and Supplementary Table 2) that include comprehensive results for all evaluated devices, including Jetson TX2. We also better clarified which device is used in the caption of each figure/table.

- **Comment 5:** Table 2 is extremely terse. A different way to visualize these data could help readers better understand the benefits of the proposed approach. Additionally, the observed out-of-memory issue could likely be avoided by using FP16 or INT8 quantization.
- **Response 5:** We thank the reviewer for the constructive suggestions. In the revised manuscript, we have replaced the original Table 2 with a more illustrative visualization (Figure 4) to better highlight the benefits of our approach. The original Table 2 has been refined and relocated to Supplementary Table 3. Relevant figure and table can be found under Response 2.

  Regarding the out-of-memory issue, we agree that using lower-precision formats such as INT8 can help mitigate the OOM issues. We have applied quantization to the baselines accordingly. The updated results of all baselines are now included in the refined Supplementary Table 3, providing comprehensive numerical comparisons.

- **Comment 6:** The section "Significance of Key Designs" is introduced before sufficiently explaining the key designs.
- **Response 6:** We thank the reviewer for the helpful observation. In the revised manuscript, we have expanded the introduction section and added an illustrative figure and a new subsection in method section to explain the key designs in detail (Figure 3, lines 83–118, 197–249) before presenting their significance in the evaluation section.

- **Comment 7:** Figure 7 presents the inference time without specifying the target devices or the quantization method used. The search time for most thresholds is still far from real-time, reaching up to 10 seconds.
- **Response 7:** We thank the reviewer for the helpful comment. The experiment shown in Figure 7 (now reorganized as Figure 5b in the revised manuscript) was conducted on Jetson Orin with the model quantized to INT8.

  As for search time, we optimized it to be only 300ms slower than naive multimodal retrieval by using a quantized model (§ 2.5.3: lines 389–416). With quantization, the full model fits into device memory without the need for layer-wise execution, allowing queries complete within 1.5 seconds to achieve with decent retrieval accuracy. In comparison, naive multimodal retrieval takes about 1.2s to finish query embedding and embedding-matrix matching. The extra latency overhead of 300ms is acceptable given the infrequent nature of such queries and the system's advantage in achieving up to 31× improvement in the throughput of frequent data embeddings. Additionally, our design supports further query latency reduction for frequently accessed items, akin to web cookies, by bypassing redundant fine-grained embedding computations.

Revised manuscript §2.5.3 (lines 389–416) Impact of Query Latency Budge

Although query costs are negligible compared to embedding costs in the long run—since queries occur less frequently than continuous daily embeddings—they are immediately noticeable to users. Thus, we illustrate `Reminisce`'s performance under different query latency budgets in Figure 5b. During queries, the device holds the entire quantized model in memory without layer-by-layer loading. Given the infrequency of queries, the temporary memory increase is acceptable. Query latency comprises three components: query embedding, matching, and fine-grained embedding. Baseline methods with memory encoders require only the first two steps, typically taking around 1.2s. `Reminisce` takes less than 1.5s (the default latency budget used in §2.5.1) to achieve acceptable query accuracy. As shown, if the system tolerates higher query delays, performance can be further enhanced. For example, on the `FLICKR` dataset, the relative retrieval accuracy of `Reminisce` improves from 92% to 99% after refining an additional 10 candidates (≈0.2s).

Additionally, similar to web cookies [41], the query process can skip the complex fine-grained embedding when repeated, improving efficiency in multi-query scenarios where frequently queried items are retrieved faster. Once a local embedding is queried, its embedding is permanently upgraded. Under these conditions, the system becomes more efficient by skipping the fine-grained embedding process for frequently queried items.

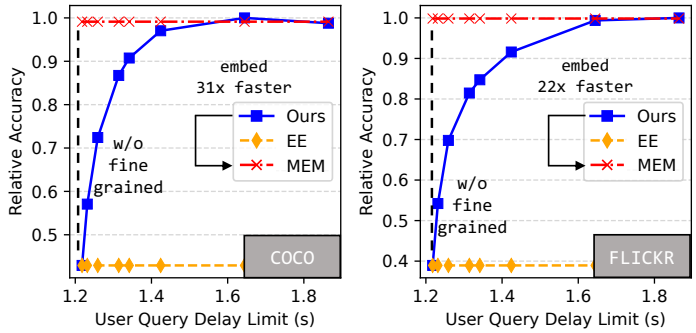


Figure 5b Performance under different query latency budgets on ORIN (INT8).

- **Comment 8:** The same concern applies to Figure 8. What device was used? What was the measurement setup?
- **Response 8:** We thank the reviewer for raising this point. Figure 8 (now updated as Figure 6 in the revised manuscript) presents the energy comparison conducted on Jetson Orin, with model weights quantized to INT8. The measurement setup is now clearly specified in the figure caption.

- **Comment 9:** In the subsection "Case Study: Twitter Meme Retrieval," the analysis is too shallow.
- **Response 9:** We thank the reviewer for the valuable feedback. In the revised manuscript, we have expanded the case study analysis in Figure 7 and §2.5.5 (lines 437–475) by incorporating a comparison with additional baselines to highlight the advantages of our system on real-world scenarios, and a more in-depth discussion of why and how our system outperforms existing methods.

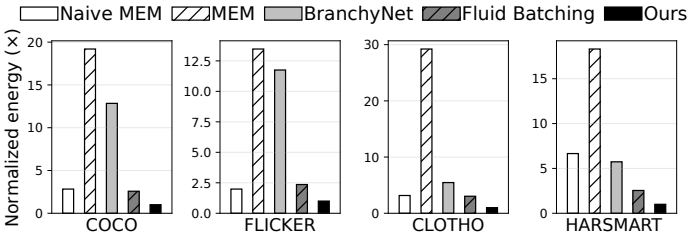


Figure 7. Comparison of energy consumption for different methods. Device: ORIN (INT8).

Revised manuscript §2.5.5 (lines 437–475) Case Study: Twitter Meme Retrieval

To demonstrate the practicality of `Reminisce` in real-world scenarios, we conducted a case study using daily surfing images and captions collected from Twitter memes as illustrate in Supplementary Figure 8. End users filtered the data to ensure privacy, and a total of 805 figures were collected to simulate 30 minutes of surfing. Our evaluation compares multiple methods—including Naive `MEM` without layer-wise execution, the `MEM` baseline, `BranchyNet`, `Fluid Batching`, and our `Reminisce` —in terms of throughput, energy, memory, and retrieval accuracy.

As shown in Figure 7, all baseline methods take over 80 minutes to complete the retrieval task on a fully utilized CPU. Naive `MEM` incurs a large memory footprint by loading the entire model at once, even with INT4 quantization. Its layer-wise execution counterpart (`MEM` baseline) reduces memory usage but significantly decreases throughput due to frequent layer-switching overhead. BranchyNet improves throughput by skipping layers but at the expense of lower accuracy. In contrast, `Reminisce` completes the same task in 28 minutes—achieving a 3× throughput improvement compared to even the strong baseline `Fluid Batching`, due to our mobile-friendly optimizations.

Our approach reduces peak memory usage by 7× compared to Naive `MEM`, lowering the footprint below 200MB. This includes a small buffer (under 50MB) for pipelined execution and temporary activations—a reasonable tradeoff for performance gains. Energy consumption is reduced by up to 4×, enabled by fewer layer computations and more efficient batching. The system also achieves higher retrieval accuracy than naive early-exit methods while maintaining an acceptable query latency of just 0.5 seconds. The additional memory overhead from batching parallelism is justified by the substantial performance improvements.

These quantitative improvements—from faster processing and lower resource consumption to robust retrieval performance—demonstrate that `Reminisce` is highly practical for deployment in mobile scenarios, where computational efficiency and low-latency requirements are critical.

- **Comment 10:** Unclear sentence: "The generated embeddings can be easily retrieved to help mobile users remember and recall information when needed." The meaning should be clarified.
- **Response 10:** We have clarified the meaning in the revised manuscript (line 3–6).

Revised manuscript (line 3–6)

Once generated, these embeddings enable mobile users to quickly retrieve relevant information, effectively augmenting their memory.

- **Comment 11:** Typo: "Based on the trace statistics in §, typical" – The reference format needs correction.
- **Response 11:** Thank you for kind reminder. We have fixed this format typo in the revised manuscript

Revised manuscript (line 430)

Based on the trace statistics in §2.1, typical. . .

- **Comment 12:** Typo: "This method outperforms fixed early-exit baselines, as will be shown in §." – The section reference should be properly formatted.
- **Response 12:** Thank you for kind reminder. We have fixed this format typo.

Revised manuscript (line 665)

This method outperforms fixed early-exit baselines, as shown in Figure 5a.

RESPONSES TO REVIEWER #2

- **Comment 1, Summary:** The manuscript presents a valid contribution to the field of multimodal embedding, specifically focusing on efficient on-device computation for resource-constrained devices, including mobile. The proposed system, Reminisce, introduces a novel approach that enhances retrieval efficiency while maintaining computational feasibility. The emphasis on hardware-friendly design is particularly valuable, ensuring practical deployment of such models on edge devices and closer results to real-world scenarios. Additionally, the study demonstrates transparency regarding resource consumption, including memory and disk usage.
- **Response 1:** Thank you for your positive evaluation and thoughtful suggestions. We are pleased to contribute practical advancements to the nature research community.

- **Comment 2:** What are the noteworthy results?
  The work achieves significant gains in energy efficiency and throughput, making it well-suited for real-world deployment on resource-constrained edge devices. The evaluation on physical (real) hardware further strengthens its practical relevance and reliability.
- **Response 2:** Thank you for recognizing the significance and thoroughness of our results.

- **Comment 3:** Will the work be of significance to the field and related fields? How does it compare to the established literature? If the work is not original, please provide relevant references.
  Yes, this work is significant to the field. It enhances multimodal embedding with a strong focus on hardware efficiency for edge (and mobile) devices. By optimising retrieval in constrained environments, it extends real-world applicability beyond prior research.
- **Response 3:** Thank you for acknowledging the significance of our work and our advantages over prior research.

- **Comment 4:** Does the work support the conclusions and claims, or is additional evidence needed?
  Yes. However, a clearer explanation of the caching invalidation strategy is necessary, as multiple caching mechanisms are involved (as pointed out in Figure 11).
- **Response 4:** Thank you for your valuable suggestions. We have added a separate paragraph (lines 776–787) and an illustrative workflow (Figure 9) to clarify the cache invalidation strategy depicted in original Figure 11 (now Figure 3a in the revised manuscript).
  In summary, all caches are invalidated immediately after assisting with updates to embeddings of higher granularity. During the multimodal data embedding phase, intermediate superficial embeddings are invalidated immediately after obtaining the corresponding coarse-grained embeddings. Coarse-grained embeddings are invalidated immediately after their first use in queries, i.e., when refined into fine-grained embeddings. During the query phase, coarse-grained query embeddings are cached for speculative retrieval and reused when computing fine-grained query embeddings for final retrieval.

Revised manuscript line 776–787

To efficiently manage intermediate activations and avoid resource waste from stale data, we adopt a cache invalidation strategy as shown in Figure 9. During offline embedding phase, intermediate activations from superficial embeddings are temporarily stored in RAM to compute coarse-grained embeddings. After each batch, these cached activations are sequentially invalidated from RAM. Coarse-grained intermediate activations are subsequently stored on disk, which has fewer constraints compared to RAM (see Supplementary §F for details). At query phase, cached embeddings matching the incoming query are loaded to compute fine-grained embeddings and are promptly invalidated afterward.

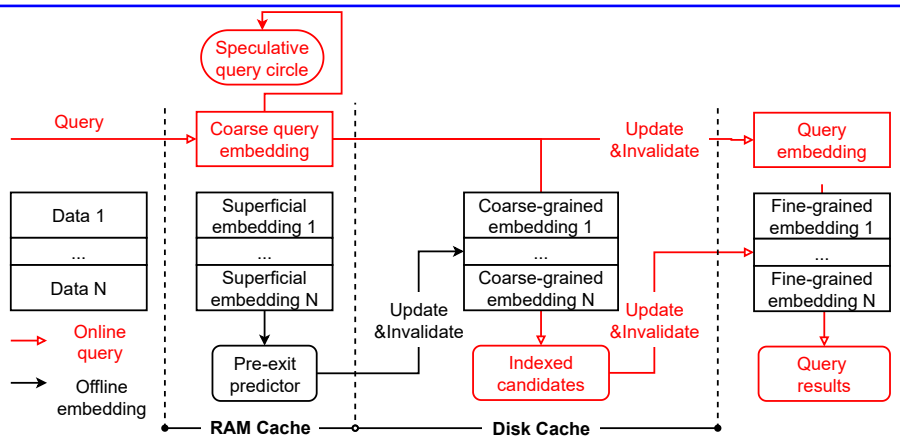


Figure 9: Invalidation strategy of Reminisce.

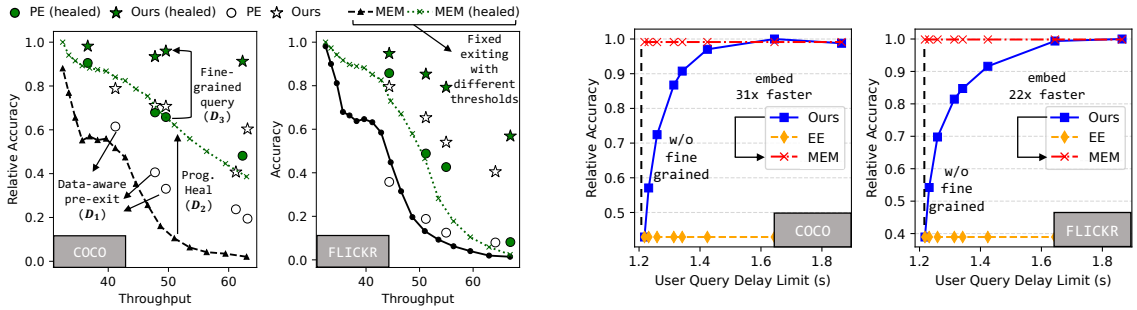
- **Comment 5:** Are there any flaws in the data analysis, interpretation, and conclusions? Do these prohibit publication or require revision?
  No.
- **Response 5:** Thank you for your acknowledgment.

- **Comment 6:** Is the methodology sound? Does the work meet the expected standards in your field?
  The methodology is appropriate and meets field standards. The approach to on-device computation is particularly strong, as it reduces energy consumption while maintaining retrieval performance, and the motivation (privacy concern when using Cloud solutions) is solid.
- **Response 6:** Thank you for acknowledging our contribution and motivation. We hope our work will have the opportunity to achieve broader impact through publication in *Nature Communications*.

- **Comment 7:** Is there enough detail provided in the methods for the work to be reproduced?
  Most aspects are well-described, but the caching invalidation mechanism should be clarified for readers.
- **Response 7:** Thank you for your acknowledgment and helpful suggestion. We have clarified the caching invalidation mechanism further in lines 776–787 and Figure 9. Please refer to Response 4 for more details.

- **Comment 8, Final Recommendation:**
  The study is well-executed and makes a valuable contribution to the field. Addressing minor issues related to caching, terminology, and figure corrections will further enhance the clarity of the manuscript.
  Figure corrections are: Figure 6 ("Ous" - > "Ours"). Figure 3b requires clarification in the legend regarding the meaning of the black and white bars colours.
  Minor terminology inconsistencies: the misuse of "backend/frontend" where "background/-foreground" would be more appropriate. Similarly, "backward" is incorrectly used instead of "background" in the appendix.
  Additionally, some redundant sentences in the appendix, such as "This causes the application to be easily killed by the OS due to memory pressure," should be removed to improve readability.
  With these refinements, I strongly support its publication.

- **Response 8:** Thank you very much for your strong support and detailed suggestions. We have addressed all the issues mentioned:
  - The legend of Figure 6 (ablation studies, now re-located as Figure 5a in the revised manuscript) has been corrected.
  - The legend of Figure 3b (preliminary experiments on battery consumption, now reorganized as Figure 2c) has been clarified. The white bars represent GPU power consumption, while black bars represent CPU power consumption. This demonstrates that GPU's higher throughput comes at the expense of greater power usage, highlighting the necessity of our approach.
  - Terminology consistency regarding "background/foreground" has been refined throughout the manuscript.
  - Many sentences in the appendix have been revised to improve readability.



(a) Impact of key designs. (b) Impact of query budget.

Figure 5: Performance analysis of Reminisce's key designs and query latency impact on ORIN (INT8). (a) Throughput-to-accuracy trade-off with and without Reminisce's key **D**esigns$_{(1,2,3)}$. PE refers to pre-exited coarse-grained embeddings without fine-grained upgrading during the query phase. (b) Performance under different query latency budgets.

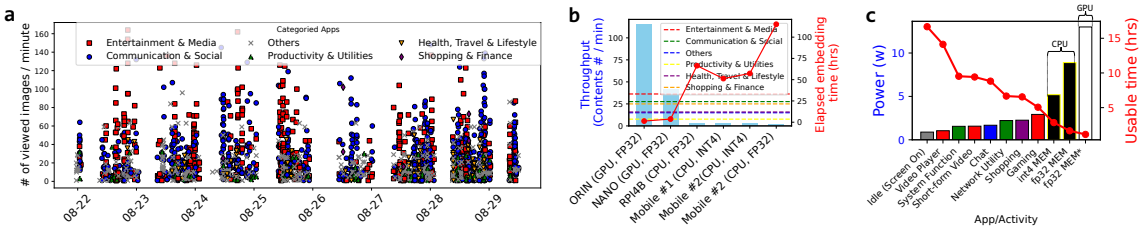


Figure 2: Motivation of multimodal embedding on mobile devices and its challenge. (a) Viewed image trace of mobile users. (b) MEM inference speed across different devices compared to the average image generation speed of various mobile applications. (c) MEMs rapidly drain the battery of the test mobile phone. * indicates GPU inference power consumption of the Jetson ORIN.

- **Comment 9, Reviewer #2 (Remarks on code availability):**
  My assessment of the code is that it is a usable resource for the community, with clear details on installation and usage of the available scripts. It also considers important (open-source) OS details, such as a clear definition of its OS license.
- **Response 9:** Thank you for acknowledging our open-source efforts. We hope that our codebase facilitates and encourages further research in this direction.