MetaSight: See, Streamline, Meta-Evolve — A Super Efficient Multimodal Agent that Evolves at the Edge

Anonymous Author(s)

Abstract

Multimodal large language models (MLLMs) are deployed today mostly as static endpoints with hard budgets: every additional video frame and prompt token costs latency and dollars, and the model has no mechanism to learn from the questions it gets wrong. We present MetaSight, a self-evolving multimodal agent that addresses both via *hybrid encoding* across three layers: a cascaded edge-side frame gate, hot/cold skill injection with top-k retrieved reasoning skills, and memory routed into a skill evolver so each retrieved exemplar reshapes the skill bank that serves every future question, rather than being concatenated alongside skills into the per-question prompt as in prior memory-augmented agents. Across 4 video-QA benchmarks with 2 VLM families: Gemini 3 Flash and GPT-5.2, MetaSight cuts per-question API cost by an average −98% versus full-frame upload (peak −99.3% on Video-MME long) and by −25.9% over the offline uniform 8 frame ceiling at the same evolved skill bank configuration, while boosting accuracy on most settings, *e.g.*, an average +3.85% and a peak +15.80% on EgoSchema using Gemini 3 Flash. When testing on a matched frame budget with the offline uniformly 8 frames per video, our cascade+uniform-filling variant still beats the straightforward uniform-8 upper-bound on almost all benchmarks (*e.g.*, 67.4% vs. 65.7% on average with `FullEvo`), and our offline-best configuration with the full evolution passes Gemini 1.5 Pro on EgoSchema with a smaller backbone. These properties make MetaSight a natural fit for live edge applications such as AI glasses, where the cascade reduces a 1-hour streaming session from ∼3,600 API uploads down to only 5–20 calls.

Keywords: Self-Evolve, Efficiency, Multimodal Agent

ACM Reference Format:
Anonymous Author(s). 2018. MetaSight: See, Streamline, Meta-Evolve — A Super Efficient Multimodal Agent that Evolves at the Edge. *Proc. ACM Meas. Anal. Comput. Syst.* 37, 4, Article 111 (August 2018), 13 pages. https://doi.org/XXXXXXX.XXXXXXX

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM 2476-1249/2018/8-ART111
https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Multimodal large language models (MLLMs) such as Gemini 3 [12], GPT-5 [19], and Claude [1] have achieved strong accuracy on video question answering: when given the entire clip and a sufficiently long prompt, frontier models score in the 70–80% range on standard MC benchmarks. The cost of this paradigm is buried in two assumptions that hold for offline benchmarks but break the moment the system is deployed — (a) the entire video is available at query time, and (b) the model is the same at the end of the day as it was at the beginning. Production video-QA agents, from cloud assistants to wearable AI glasses, violate both: frames arrive serially over a stream of unknown length, and the agent is expected to get progressively better at the questions it actually receives.

These two assumptions have so far been relaxed only in isolation. On the efficiency side, frame-selection systems [14, 31, 34, 42] compress or sub-sample the input video, but each of them requires offline access to the full clip and none addresses the prompt-side cost: any reasoning context the practitioner injects scales linearly with bank size and is re-sent to the model on every query. On the adaptation side, skill libraries for LLM agents [32, 33, 36, 46] distil reusable behavioural rules from past failures and inject them at inference, demonstrating that frontier-model accuracy can be lifted *without weight updates*. But all of these systems are text-only — their multimodal extension is the open problem, and it raises a new constraint that the text-only setting does not face: how do we encode a growing bank of skills and memory so that the bank itself does not become a token bottleneck?

We present MetaSight, a self-evolving multimodal agent built around a single design principle: *hybrid encoding* at every stage of the pipeline, where information is processed at multiple fidelity tiers and only a selected portion gets the expensive treatment. **(i)** The frame stream passes through a cascaded gate at three encoder costs — perceptual hash, 128-dim CPU encoder, adaptive change-gate — running at >100 fps on a single CPU thread, so the cloud VLM only ever sees salient transitions. **(ii)** The reasoning bank is encoded into a hot top-k tier (with full skill content) plus a cold catalogue tier (with just name + one-line description), keeping per-question prompt cost flat as the bank grows during online evolution. **(iii)** Memory is encoded as confidence-gated dense retrievals that condition the *offline evolver* rather than the per-question VLM call, shielding the high-frequency call from low-signal exemplars. The bank itself meta-evolves: an

LLM evolver synthesises new skills from observed failures, and a per-skill utility tracker continually prunes the bank, with no weight updates to the VLM at any stage.

We evaluate our method across 4 benches with 2 VLM families: two egocentric benches (EgoSchema, EgoPlan-Bench) and two general-video benches (Video-MME long, NextQA), tested on Gemini 3 Flash and GPT-5.2 to measure cross-VLM transfer of the memory-driven evolved skill bank. The streaming-deployable configuration is positive on most settings (*e.g.*, 6 of 8 experiments) with an average lift of +3.85% and a peak of +15.80% on EgoSchema using Gemini 3 Flash. Moreover, we consider an offline upperbound with 8 uniformly sampled frames per video and the proposed evolved skills, which adds another +3.50% to +13.00% on top. On efficiency, per-question API cost drops by −25.9% on average against a matched-method offline ceiling and by −98% against full-frame upload (peak −99.3% on V-MME long); even at matched K=8 frame budget, our cascade-fill variant still beats simple uniform-8 + `FullEvo` (*e.g.*, +1.95% on EgoPlan-Bench, +1.50% on NextQA). Beyond the headline scorecard, ablations surface findings of independent interest: routing memory through the offline evolver outperforms per-question concatenation by an average +2.05% (peak +3.80% on EgoSchema with Gemini 3 Flash) and is especially helpful for longer video content; skill and memory activations are sparse (top-3 of a 40–67-skill bank, ∼1 memory retrieval per 21–59 questions); and the lift is conditional on whether the target VLM exhibits the bank-evolved-against failure modes rather than raw capability (+15.80% on Gemini 3 Flash vs. +4.00% on the stronger GPT-5.2 EgoSchema row).

2 Related Work

Skill-based and memory-augmented agents. A line of work augments agents with reusable skill libraries or external memory to improve performance without modifying model weights. Reflexion [29] stores verbal self-reflections in an episodic buffer; Voyager [33] incrementally builds a library of executable code skills from successful episodes; ExpeL [46] and Agent-KB [32] distill cross-task experience into natural-language rules. Memory systems include MemGPT [21], Generative Agents [23], Mem0 [8], and MemEvolve [43]. A shared limitation is that the skill library is treated as a static artefact, not coordinated with weight optimisation. MetaClaw [36] addresses this by coupling skill evolution with RL training; MetaSight keeps the coupling but freezes the policy weights, relying instead on per-skill utility tracking and bank hygiene to keep the library quality-controlled across long evolution histories.

Continual and meta-learning. Meta-learning frames learning as optimisation for fast adaptation to new tasks. RL^2 [9], PEARL [24], and ProMP [27] demonstrate fast adaptation in robotic control with low-dimensional action spaces.

Table 1. Selected-frame MLLMs. Selector is the frame selection method. Online indicates if the method keep/skip decision frame-by-frame as frames arrive. Edge-CPU: selector runs on-device with no GPU. And existing multimodal agents are unable to evolve natively.

Method	Selector	Online	Edge-CPU	Evolve
LLoVi [42]	uniform chunks	no	no	–
MovieChat [31]	sliding window	no	no	–
MA-LMM [14]	trained mem-Q	no	no	–
VideoAgent [34]	LM planner	no	no	–
Frame-Voyager [40]	LM planner	no	no	–
SeViLA [39]	BLIP-2 locator	no	no	–
VideoTree [35]	LM tree-descent	no	no	–
TimeChat [26]	uniform	no	no	–
VTimeLLM [15]	uniform	no	no	–
Video-LLaMA [44]	uniform	no	no	–
LLaVA-Video [45]	uniform	no	no	–
MetaSight	**cascade (heuristic)**	**yes**	**yes**	**yes**

Continual learning studies sequential task adaptation without forgetting [4, 16, 41]. Online meta-learning relaxes the offline assumption. MetaClaw [36] extends this to LLM agents in a non-stationary task stream, with strict support/query separation and a versioning protocol; MetaSight inherits the protocol structure and applies it to video-QA failure distillation, with per-skill utility tracking serving the role of MetaClaw's stale-reward filtering.

Selected-frame and efficient video VLMs. A growing line of work selects a small subset of frames before invoking the VLM, but the prevailing approach puts an LM *in the per-frame selection loop* or assumes offline access to the full clip. LLoVi [42] extracts per-chunk captions and chains them through an LLM. MovieChat [31] and MA-LMM [14] use memory-augmented sliding windows or learned memory queries to compress long video. VideoAgent [34] and Frame-Voyager [40] treat keyframe selection as an LLM-planning problem, iteratively scoring candidate frames against the question. SeViLA [39] self-chains a BLIP-2-style locator to a BLIP-2 answerer, also requiring full-clip access. VideoTree [35] builds a hierarchical tree of frames and adaptively descends query-relevant branches. TimeChat [26] and VTimeLLM [15] prefix time tokens to a uniformly-sampled frame stack to enable temporal grounding, but inherit the uniform-sampling cost. Vid2Seq [38], Video-LLaMA [44], and LLaVA-Video [45] operate on uniform-sampled stacks at fixed budgets. Table 1 summarizes the comparison: MetaSight's cascade is the only mechanism in this set that (a) runs CPU-only on the edge, (b) makes its decision frame-by-frame as frames arrive, and (c) does not require any LM in the selection loop — a point the rest of the literature has not occupied because they are designed for offline benchmarks rather than streaming deployment. Orthogonally, none of these methods evolve a reasoning bank or memory store across the task stream; the prompt-side cost in each is fixed at deployment time.

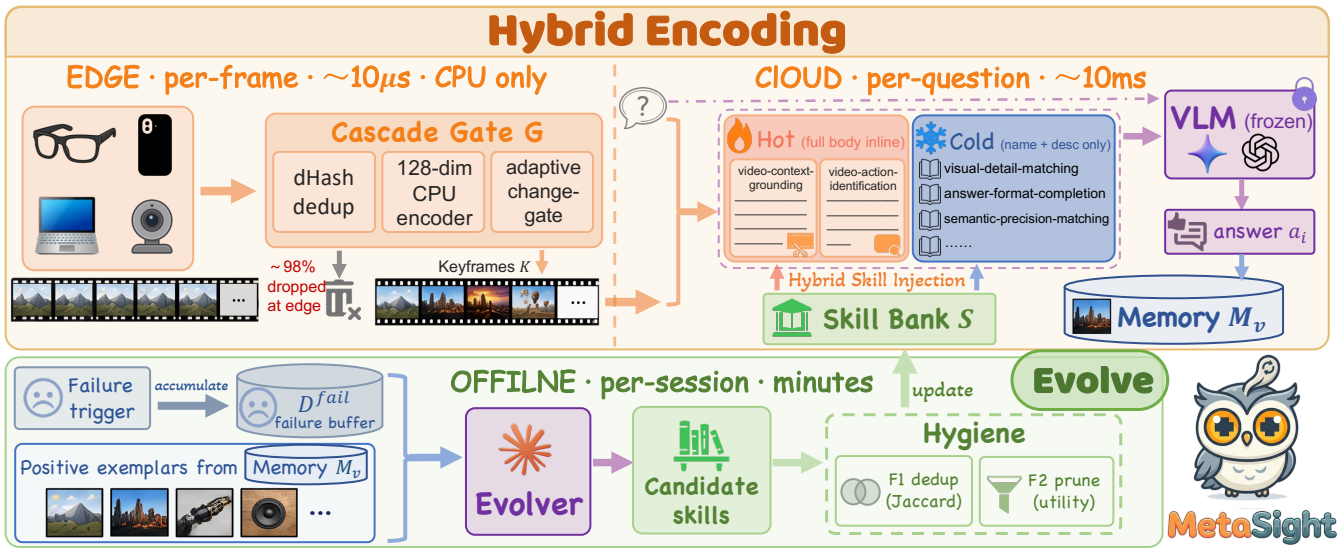

Figure 1. MetaSight pipeline. A two-tier system with hybrid encoding and meta-evolve. An on-device cascade gate encodes frames *per-frame*, and a memory-augmented evolver paired with a hot/cold skill injector evolves the language-layer scaffolding *per-question* and *per-session*.

Egocentric and general video QA benchmarks. We evaluate our method on both egocentric and general multimodal data. EgoSchema [18] is the standard 5-way MC benchmark for 3-min ego clips; EgoPlan-Bench [5] targets ego planning from a single observation frame. Ego4D [13], Charades-Ego [30], EgoThink [7], and VidEgoThink [6] provide complementary task formats. For general video, Video-MME [10] covers 12 task types across short/medium/long durations, NextQA [37] tests causal/temporal reasoning on ~30 s YouTube clips, and IntentQA [17] targets intent reasoning. We report results of MetaSight on EgoSchema, EgoPlan-Bench, Video-MME long, and NextQA.

Tool registries and hybrid injection. Toolformer [28] trains LLMs to call tools; MCP [2] standardises tool advertising between models and clients. Our hybrid hot/cold skill injection is conceptually adjacent — we serve a small "hot" set inline and a "cold" catalogue with on-demand body fetch, but apply this to *reasoning* skills rather than tool definitions, and the choice between hot/cold tiers is made by retrieval against the current question rather than declared statically.

3 Method

To address the visual redundancy on the input side and frozen scaffolding on the reasoning side, our MetaSight composes three filtering stages, each operating at its own timescale: an edge-side cascaded gate G that triages frames *per-frame*, a hot/cold skill injector that triages the bank S *per-question*, and a memory-augmented evolver that distils new entries into S *per-session* from a confidence-gated episodic store M_v. The first stage cuts the visual cost; the latter two evolve the language-level scaffolding $\{S, M_v\}$ over the task stream. VLM weights θ are never updated. We present MetaSight pipeline of hybrid encoding and meta-evolve in Figure 1.

3.1 Preliminaries and Notation

We denote the multimodal model to be evolved as $M = (\theta, S, M_v, G)$, with θ the (frozen) weights of a cloud VLM, $S = \{s_1, \ldots, s_K\}$ a library of language skills, M_v an episodic memory store indexed by dense sentence embeddings, and G an edge-side cascade visual encoding gate we proposed. For a question q over a stream of frames $\mathcal{F}$, the answer a is generated by:

$$a \sim \pi_\theta\big(\cdot \mid q,\ G(\mathcal{F}),\ \mathrm{Ret}_S(q,k),\ \mathrm{Cat}(S),\ \mathrm{Ret}_{M_v}(q)\big), \qquad (1)$$

where $\mathrm{Ret}_S(q,k)$ returns the top-k retrieved skills (the hot tier), $\mathrm{Cat}(S)$ produces the name-and-description cold catalogue, and $\mathrm{Ret}_{M_v}(q)$ retrieves confidence-gated memory snippets. The three components $\{G, S, M_v\}$ are updated at three qualitatively-different timescales:

Per-frame (~10 μs, edge). G inspects each arriving frame and emits a MAJOR / MINOR / SKIP verdict. CPU-only.

Per-question (~10 ms, cloud). Ret_S and Ret_{M_v} rank the bank against q; the top-k skills are inlined, the rest catalogued.

Per-session (minutes, offline). After every N_{evo} failures the LLM evolver $\mathcal{E}$ analyses failure trajectories and proposes new skills, $S \leftarrow S \cup \mathcal{E}(S, D^{\text{fail}}, M_v)$; bank-hygiene filters F_1, F_2 maintain bank quality.

Algorithm 1 MetaSight streaming inference loop.

Require: Meta-model $M = (\theta, S, M_v, G)$; question stream $\{q_i\}$; frame stream $\{f_t\}$; evolve threshold N_{evo}; hybrid top-k.

$K \leftarrow \emptyset$; $H \leftarrow \emptyset$; $D^{\text{fail}} \leftarrow \emptyset$
▷ per-frame, sub-ms
for each frame f_t from edge **do**
  $h \leftarrow \text{dHash}(f_t)$
  if $\min_{h' \in H} \text{Hamming}(h, h') \le 6$ **then**
    continue ▷ near-duplicate
  end if
  $H \leftarrow H \cup \{h\}$; $e \leftarrow \text{LightweightEncoder}(f_t)$
  $g \leftarrow \text{ChangeGate}(e; \tau_{\text{major}}, \text{decay})$
  if $g = \text{MAJOR}$ **then**
    $K \leftarrow K \cup \{f_t\}$
  end if
end for
▷ per-question, ~10 ms
for each question q_i **do**
  $S_i^{\text{hot}} \leftarrow \text{Ret}_S(q_i, k)$
  $S_i^{\text{cold}} \leftarrow \text{Cat}(S \setminus S_i^{\text{hot}})$
  $m_i \leftarrow \text{Ret}_{M_v}(q_i)$ ▷ cosine ≥ 0.55
  $a_i \leftarrow \pi_\theta(\cdot \mid q_i, K, S_i^{\text{hot}}, S_i^{\text{cold}})$ ▷ cloud VLM call
  $r_i \leftarrow \text{Score}(a_i, q_i)$
  $\text{UpdateUtility}(S_i^{\text{hot}}, r_i)$
  if $r_i = 0$ **then**
    $D^{\text{fail}} \leftarrow D^{\text{fail}} \cup \{(q_i, a_i, m_i)\}$
  end if
  if $|D^{\text{fail}}| \ge N_{\text{evo}}$ **then**
    ▷ per-session, minutes
    $\Delta S \leftarrow \mathcal{E}(S, D^{\text{fail}}, m_i)$ ▷ evolver fires, fusion
    $\Delta S \leftarrow \text{F1Dedup}(\Delta S, S)$ ▷ Jaccard ≥ 0.5 rejected
    $S \leftarrow S \cup \Delta S$; $D^{\text{fail}} \leftarrow \emptyset$
  end if
  if $i \bmod 100 = 0$ **then**
    $S \leftarrow \text{F2Prune}(S)$
  end if
end for

The full cascade encoding and evolution loop is summarized in Algorithm 1.

3.2 Per-frame: Cascaded Encoding Gate

Most frames within a contiguous window of a streaming video are visually redundant, and cloud-VLM cost grows linearly with each forwarded frame. We therefore filter the stream content-aware on the edge, forwarding only the salient transitions with a decision rule that consumes no future frames — ruling out any selector that needs the full clip up front. Concretely, the visual encoding gate G maps each arriving frame f_t to a verdict $g_t \in \{\text{MAJOR}, \text{MINOR}, \text{SKIP}\}$ from $f_{1:t}$ alone: MAJOR frames cross the major-change threshold and are forwarded to the cloud VLM as keyframes, MINOR frames cross only a lower threshold and update the rolling reference for subsequent comparisons but are *not* uploaded, and SKIP frames fall below both thresholds and trigger no action at all. The keyframe set forwarded to the cloud is therefore $K = \{f_t : g_t = \text{MAJOR}\}$, while MINOR and SKIP frames are discarded at the edge. G composes three $O(1)$-per-frame stages, applied in order:

- **Perceptual hash (dHash).** Hamming-distance dedup against a rolling buffer of recent hashes rejects bit-exact and near-exact duplicates, catching camera shake, identical-scene re-frames, and stationary periods.
- **Lightweight encoder.** A 128-dim CPU-only feature combining HSV histogram, luminance, edge density, and texture statistics. No deep network, no GPU; produces a scene-similarity vector usable with cosine distance.
- **Adaptive change gate.** Compares the current encoded frame against a rolling reference and emits MAJOR / MINOR / SKIP verdicts using thresholds that adapt with temporal decay, so the gate fires reliably even on slow-moving scenes and stationary cameras.

Because g_t depends only on $f_{1:t}$, the cascade runs on a live stream of unknown length without buffering or replay. Only frames in K are forwarded to the cloud VLM; everything else is discarded at the edge.

3.3 Per-question: Skill Bank with Hybrid Hot/Cold Injection

Agent scaffolds empower LLM agents without weight updates, but each injected skill costs prompt tokens at every query, and once the skill bank S exceeds tens of entries full-inline injection saturates the prompt context and obscures task-specific signal. We therefore inject the skill in two tiers. Each text skill $s_j \in S$ is a short markdown card with a name, a one-line description (the retrieval key for skill evolve), a numbered procedural body, and an explicit anti-pattern section, and S is initialized with a small seed bank of K_{seed} cross-cutting visual-reasoning patterns that are bootstrapped from a held-out probe run of the same skill evolver $\mathcal{E}$, and grows during deployment thereafter. We treat each s_j as an *implicit preference rule* rather than a procedural recipe.

To adopt skills dynamically, for each incoming question q, a sentence-transformer embedding ranks S against the question text. The top-k skills $S^{\text{hot}} = \text{Ret}_S(q, k)$ are inlined as *hot* bodies into the system prompt; the remaining skills become a *cold catalogue* $S^{\text{cold}} = \text{Cat}(S \setminus S^{\text{hot}})$ of name-and-description pairs only, with bodies fetchable on demand if the model decides it needs them. Per-question prompt cost is therefore bounded by k rather than $|S|$, decoupling injection cost from bank growth during online evolution.

3.4 Per-session: Memory-augmented Meta-evolution

In real-life scenarios, as use cases become complex, a static S cannot adapt to bench-specific failure modes the user did

Table 2. Main results across 4 benchmarks × 2 VLMs. **Cascade** columns report 6 streaming-deployable configurations; **Uniform-8 plain** is the offline baseline upper-bound. We mark best per row across the cascade columns in **bold**.

		Cascade (streaming-deployable)						Uniform-8
Bench	VLM	Plain	Seed	+Evolve	+SkillMemCat	+Evo+SkillMemCat	FullEvo (Δ)	Plain
EgoSchema	Gemini	52.60	67.20	68.00	65.20	64.60	**68.40** (+15.80)	60.60
	GPT-5.2	64.00	66.60	66.20	67.20	65.60	**68.00** (+4.00)	70.60
V-MME long	Gemini	60.33	61.56	61.33	62.78	62.56	**64.22** (+3.89)	61.44
	GPT-5.2	55.89	54.00	52.22	54.67	52.78	**55.89** (0.00)	58.78
EgoPlan-Bench	Gemini	24.62	**30.80**	29.93	28.31	30.04	28.85 (+4.23)	37.96
	GPT-5.2	28.42	28.74	28.31	28.09	28.85	**29.39** (+0.97)	43.06
NextQA	Gemini	72.70	75.10	73.90	75.50	**75.70**	74.50 (+1.80)	77.70
	GPT-5.2	73.20	72.00	72.30	70.90	72.50	**73.30** (+0.10)	78.90

not anticipate, while a naive LLM evolver invoked on raw failures introduces two new failure modes of its own: it may emit narrow recipes that overfit to the triggering failure, and S may bloat with near-duplicates over long evolution histories. We therefore want an evolver $\mathcal{E}$ that is conditioned on positive examples so it abstracts from successful patterns rather than memorizing the failure surface, and a bank that self-cleans as it grows.

Unlike prior memory-augmented agents [21, 29, 36] that concatenate retrieved memory directly into the per-question prompt, we route memory into the offline evolver — so each retrieved exemplar reshapes the skill bank that serves every future question rather than competing with the visual context for one-shot attention. An auxiliary episodic memory store M_v records correctly-answered questions as dense embeddings (sentence-transformer), with queries combining question and option text and retrieval confidence-gated by cosine similarity so only on-topic exemplars surface. Let D^{fail} accumulate scored failures since the last evolve. When $|D^{\text{fail}}| \geq N_{\text{evo}}$ the evolver fires: $\Delta S = \mathcal{E}(S, D^{\text{fail}}, M_v)$ retrieves the top exemplars from M_v that match the failure context, conditions its skill-synthesis prompt on them, and proposes new entries. We expose memory → VLM concatenation as an optional secondary lever (§4.2 validates that fusion beats concatenation on 3 of 4 Gemini experiments).

Two filters keep the skill bank S quality-controlled across long evolution histories: **(F1)** a token-Jaccard dedup at evolve-time rejects entries in ΔS whose names overlap heavily with existing $s_j \in S$, and **(F2)** a per-skill utility tracker logs each s_j's hit rate on scored answers and periodically prunes skills whose accuracy lags the bank mean. Together they make the per-session evolution loop "self-healing" rather than monotonically blowing.

4 Experiments

4.1 Setup

Multimodal Language Models. We test two cloud VLMs — Gemini 3 Flash [12] (lightweight) and GPT-5.2 [20] (one of the strongest available) — with Claude Haiku 4.5 [3] as the offline evolver and `all-MiniLM-L6-v2` [25] as the memory encoder. **Cascade gate.** Frames arrive at 1 fps with up to 8 keyframes per question; the major-change threshold τ_{major}=0.30 relaxes from 4 s of stability (`decay_start`) toward a floor of 0.3 (`decay_floor`), and a 10 s silence ceiling forces a MAJOR verdict so the gate also fires on stationary scenes. **Evolve trigger.** N_{evo}=15 failures, with K_{seed}=12 skills in the bootstrapped seed bank. **Variants.** We compare five configurations sharing the same cascade gate: `Plain` (bare VLM, no scaffolding); `Seed` (the K_{seed}-skill seed bank injected per question); `+Evolve` (seed plus the evolver firing every N_{evo}=15 failures, no memory); `+SkillMemCat` (seed plus top-3 confidence-gated memory exemplars concatenated alongside the skills into the per-question VLM prompt, no evolver); and `FullEvo` (all three, with memory routed to condition the evolver instead of the per-question prompt — the §3.4 design choice).

Benchmarks. We evaluate MetaSight on two ego-view and two general video-QA datasets to simulate the use of real-life scenarios and to probe in more general multimodal situations. For the ego-view datasets, we use EgoSchema [18] with 500 instances, 3-min clips on average to test long-horizon ego activity recognition, and EgoPlan-Bench [5] with 923 entries and single observation frame for planning tasks. As for the general video QA, we test on Video-MME long [10] with 900, and 30+ min video content to test the reasoning at a long clip duration, as well as NextQA [37] with 1000 instances and ~30s videos for short-clip causal/temporal reasoning.

4.2 Accuracy Results and Analysis

Table 2 presents the main scores for 6 streaming-deployable configurations alongside the plain baseline at uniform-8

Table 3. `FullEvo` on top of uniform-8 sampling delivers a positive lift across all four benches using Gemini 3 Flash.

Bench	U-8 plain	U-8 + `FullEvo`	Δ
EgoSchema	60.60	**73.60**	+13.00
V-MME long	61.44	**66.78**	+5.34
EgoPlan-Bench	37.96	**50.11**	+12.15
NextQA	77.70	**81.20**	+3.50

frame budget, where we uniformly extract 8 frames in the video for question-answering. The `FullEvo` consists of the skill evolve and memory → evolver fusion; injection mode with memory-guided skill evolution instead of simple concatenation. The Uniform-8 plain column is included as an offline reference: it shows what the same VLM achieves with offline access to all 8 sampled frames but no method scaffolding added.

MetaSight Scores at the Top. In Table 2, MetaSight's `FullEvo` achieves an average performance boost of +3.85% over the Plain baseline and a peak of +15.80% on EgoSchema with Gemini 3 Flash. Notably, Gemini 3 Flash yields at least 4% lift at the streaming budget on two egocentric benchmarks (EgoSchema +15.80%, EgoPlan-Bench +4.23%), demonstrating its advantage in real-world applications like personalized AI glasses.

To better understand the role of each function, we walk through each component in Table 2, summarising lifts as the four-bench average per VLM. The initial seed skill bank alone (`Seed`) lifts Gemini's Plain by +6.10%, suggesting that the bootstrapped seed addresses the dominant Gemini failure modes. However, the three separate components: adding the skill evolver (`+Evolve`, +5.73% over Plain), memory concatenation alone (`+SkillMemCat`, +5.39%), or evolver with concatenation together (`+Evo +SkillMemCat`, +5.67%) do not improve over the `Seed` baseline on Gemini or the GPT-5.2 variant, suggesting evolved skills and concatenated memory each compete with the initial seed skill bank for prompt budget when added in isolation. Only the full `FullEvo` that routes memory through the evolver instead of concatenating it, consistently beats `Seed` on both VLMs (+0.33% on Gemini, +1.31% on GPT-5.2 over Seed), and is the only variant that lifts the stronger GPT-5.2 backbone above Plain on average (+1.27% vs. −0.04 to −0.62% for the four other variants), confirming that the proposed memory routed skill evolution drives the major performance lift.

Validating `FullEvo` on the Offline Baseline. As the increased accuracy of MetaSight primarily comes from the memory-driven skill evolution we proposed, we apply `FullEvo` to the Uniform-8 offline method in Table **??** to better validate its use. Even added to the stronger baseline, our `FullEvo` delivers an additional +3.50 to +13.00% over the offline Plain baseline, peaking at +13.00% on EgoSchema (60.60 → 73.60) and +12.15% on EgoPlan-Bench. This forms an offline upper-bound (Uniform-8 + `FullEvo`) that our streaming `FullEvo` approximates within 5.20% on EgoSchema and 2.56% on V-MME long while running at a small fraction of the cost (see §4.3).

Table 4. EgoSchema leaderboard results. All baseline numbers are reproduced from cited works.

Method	Acc. (%)	Backbone
Frozen Bilinear (baseline) [18]	17.6	ResNet+BERT
LongViViT [22]	56.8	ViViT
LLoVi (GPT-4) [42]	61.2	GPT-4
LLoVi (GPT-4o) [42]	67.6	GPT-4o
VideoAgent [34]	71.3	GPT-4 + LLM planner
Gemini 1.5 Pro [11]	72.2	Gemini 1.5 Pro
MetaSight (cascade + `FullEvo`, online)	67.2	Gemini 3 Flash
MetaSight (uniform-8 + `FullEvo`, offline)	**73.6**	Gemini 3 Flash

Memory Routing Beats Concatenation. We isolate the contribution of routing memory through the skill evolver against two memory-concatenation baselines that differ only in whether the evolver is active. `+SkillMemCat` disables the evolver and concatenates retrieved memory alongside the seed bank into the per-question prompt, which matches prior memory-augmented methods. `+Evo +SkillMemCat` keeps the skill evolver running but still concatenates memory at answer time, matching `FullEvo`'s components and differs only in the memory routing mechanism. And in Table 2, `FullEvo` wins in most cases with an average lift of +2.05% and a peak of +3.80% on EgoSchema using Gemini 3 Flash, with +3.11% on V-MME long using GPT-5.2, which also confirms the direction transfers across VLM families. However, when dealing with shorter video clips (*e.g.*, EgoPlan-Bench and NextQA), Gemini 3 Flash with our `FullEvo` shows slight decrease of the performance (*e.g.*, −1.19% on average). This is because brief visual context leaves enough prompt-budget headroom that concatenated memory does not crowd the question signal. Moreover, the comparison against `+SkillMemCat` mirrors the same direction on three of four Gemini experiments (*e.g.*, average +1.73%, peak +3.20% on EgoSchema), supporting our choice to fuse memory into the offline evolver rather than concatenate it into the model prompt.

MetaSight Surpasses Frontier Baselines on EgoSchema. On the EgoSchema leaderboard (Table 4), MetaSight's streaming-deployable configuration (online) achieves 67.20%, sitting within 4.10% of VideoAgent's offline LLM-driven planner (71.30%) at a fraction of the latency since the cascade rejects ∼98% of frames before any radio is woken (§4.3). Pushing to the offline-best configuration (uniform-8 + `FullEvo`, 73.60%) edges past Gemini 1.5 Pro's 72.20% on a 4× smaller, cheaper Gemini 3 Flash backbone, and outperforms LLoVi (GPT-4o) by 6.00% (73.60 vs. 67.60). The performance boost over the per-VLM Plain baseline at uniform-8 (*e.g.*, 60.60 → 73.60 with +13.00%) further demonstrates that the gain compounds

Table 5. Per-benchmark API-cost comparison on Gemini 3 Flash. KF/Q = frames sent to the API; tok/Q = input tokens per question; $/run = Gemini 3 Flash spend across the bench at $0.30/M input + $2.50/M output. Savings are our MetaSight vs the two baselines.

Bench (n, avg dur)	Configuration	KF/Q	tok/Q	$/run	vs Full-frame	vs U-8+FullEvo
EgoSchema (3 min)	Full-frame @1 fps	~180	~192,841	$28.93	—	—
	Uniform-8 + FullEvo	8.00	13,419	$2.01	—	—
	MetaSight	2.95	9,524	**$1.44**	−95.0%	−28.4%
V-MME long (~30 min)	Full-frame @1 fps	~1,800	~1,926,361	$520.12	—	—
	Uniform-8 + FullEvo	8.00	15,818	$4.28	—	—
	MetaSight	5.41	13,420	**$3.63**	−99.3%	−15.2%
EgoPlan (~23 s)	Full-frame @1 fps	~23	~16,363	$4.53	—	—
	Uniform-8 + FullEvo	8.00	13,348	$3.69	—	—
	MetaSight	1.13	10,728	**$2.97**	−34.4%	−19.5%
NextQA (~30 s)	Full-frame @1 fps	~30	~32,429	$9.73	—	—
	Uniform-8 + FullEvo	8.00	14,025	$4.21	—	—
	MetaSight	1.51	8,207	**$2.47**	−74.6%	−41.3%
All experiments total (n=3,322 Q, 4 benches)		—	—	$563.31 / $14.19 / **$10.51**	−98.1%	−25.9%

Table 6. Parity-budget comparison (K=8, Gemini 3 Flash). Cascade-fill (cascade gate + uniform-fill of unused slots) beats uniform-8 sampling at matched frame budget.

Configuration	NextQA	EgoPlan-Bench
U-8	77.70	37.96
Cascade-fill	79.50 (+1.80)	41.54 (+3.58)
U-8 + FullEvo	81.20	50.11
Cascade-fill + FullEvo	**82.70** (+1.50)	**52.06** (+1.95)

Table 7. Skill bank composition at end-of-run, Gemini 3 Flash with full-inject FullEvo. Total = K_{seed} seed (kept) + evolved (added by the evolver). F1 = token-Jaccard dedup rejections at evolve-time. F2 = utility-tracking prune events.

Bench	Seed	Evolved	Total	F1 rejects	F2 prunes
EgoSchema	12	28	40	0	0
Video-MME long	12	45	57	0	0
EgoPlan	12	55	67	0	0
NextQA	12	32	44	0	0

when the offline budget becomes available, justifying both the streaming-deployable and offline-best in the rest of §4.

4.3 Efficiency Results and Analysis

Accuracy is one of MetaSight's two main claims; the other is that our cascade gate plus hybrid skill injection cut API cost materially against both the mainstream Full-frame @1 fps practice and the offline upperbound from §4.2. We compare frames sent to the API, input tokens, and per-run dollar spend on Gemini 3 Flash across the four benchmarks against these two baselines (Full-frame @1 fps: ship every 1 fps frame of the clip into the VLM; Uniform-8 + FullEvo: the offline upperbound from §4.2), complemented by a matched-budget accuracy comparison at K=8 frames that isolates frame-selection quality from frame-count.

Cascade Cuts API Cost by an Order of Magnitude. In Table 5, we compare our cascade-driven video encoding against two baselines on Gemini 3 Flash: the mainstream Full-frame @1 fps and the offline upperbound Uniform-8 + FullEvo. Against Full-frame @1 fps, our cascade ships 2–340× fewer frames per question and reduces dollar cost by −34.4% to −99.3% per benchmark, with an average of −98.1% over the four benches and a peak of −99.3% on V-MME long where 30-min clips would otherwise consume ~1.93M input tokens per question. Notably, the saving widens with clip duration (−95.0% on 3-min EgoSchema, −99.3% on 30-min V-MME long, −34.4% on the single-frame EgoPlan-Bench), evidencing that the cascade's dHash gate is most effective on long-form streaming workloads. Moreover, against the offline upperbound under the same skill bank configuration, our cascade still ships only 1.13–5.41 frames per question and undercuts the upperbound's cost by −15.2% to −41.3% per benchmark (average −25.9%, peak −41.3% on NextQA), while tracking its accuracy on long clips within 5.20% on EgoSchema and 2.56% on V-MME long. Overall, the cumulative spend across the four benches drops from $563.31 to **$10.51** at FullEvo, and end-to-end latency is dominated by the cloud round-trip rather than the cascade itself (< 10 ms on-device, > 100 fps on CPU; full profile in Appendix A.7).

Cascade-fill Beats Uniform-8 at Matched Frame Budget. In Table 6, we evaluate matched-budget accuracy at K=8 frames against the offline upperbound on Gemini 3 Flash. The motivation is that on short-clip benches (*e.g.*, EgoPlan-Bench and NextQA, both with ≤ 30 s clips), our cascade naturally selects only 1.13–1.51 keyframes per question, undershooting the upperbound's 8-frame budget; to isolate frame-selection quality from frame-count, we additionally test *cascade-fill*, which keeps all cascade-selected keyframes and pads up to K=8 with uniformly-sampled fillers, matching the upperbound's frame budget exactly. At this matched K=8 budget, cascade-fill beats pure uniform-8 *both* without

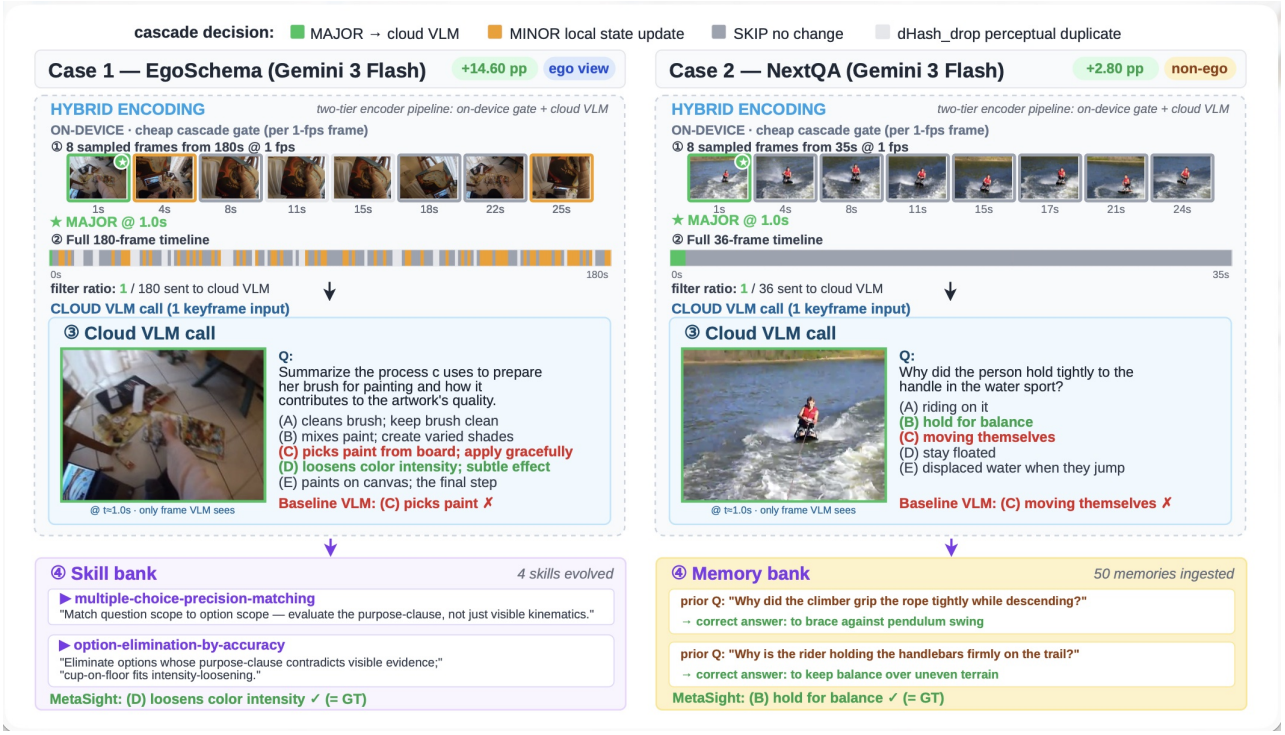

Figure 2. Two MetaSight wins on Gemini 3 Flash with efficient encoding and evolved skill and memory. Case 1 (EgoSchema, **+14.60**%): single MAJOR keyframe (1/180); evolved skills flip baseline **(C)** "picks paint" → GT **(D)** "loosens color intensity." Case 2 (NextQA, **+2.80**%): single keyframe (1/36); two memory-bank exemplars on the "hold-tight → stabilise" pattern flip **(C)** "moving themselves" → GT **(B)** "hold for balance."

skill scaffolding (+1.80% on NextQA, +3.58% on EgoPlan-Bench) and with our `FullEvo` on top (+1.50% on NextQA, +1.95% on EgoPlan-Bench), evidencing that scene-change keyframes carry more signal than evenly-spaced ones at fixed cost. Notably, cascade-fill + `FullEvo` reaches 52.06% on EgoPlan-Bench, a +14.10% lift over plain U-8 (37.96), showing that our frame-selection quality and the skill-side lift compound rather than compete.

4.4 Further Analysis

Beyond MetaSight's headline accuracy and efficiency, three operational properties decide whether the system is deployable in practice: how the proposed skill bank generalises across VLM families, the runtime behaviour of the evolving bank and memory store, and the implications for edge-device streaming deployment. We probe each in turn below.

Capability-conditional Gains. The headline lift is conditional on whether the target VLM exhibits the failure modes the bank was evolved to address, not on raw capability. `FullEvo` delivers +15.80% on the weaker Gemini 3 Flash but only +4.00% on the stronger GPT-5.2 EgoSchema row, despite GPT-5.2 starting from an 11.40% higher Plain baseline (64.00 vs. 52.60): GPT-5.2 simply does not exhibit Gemini's premature-abandonment pattern at the same rate, so the seed-bank's `answer-format-completion` skill that recovers Gemini becomes near-redundant on GPT-5.2. Across off-family backbones (*e.g.*, GPT-4o, Claude Sonnet 4.5), the lift further attenuates with the same Gemini-evolved bank, indicating that portability is bounded to the VLM family the bank was evolved against. Per-skill transfer dynamics (format-skill VLM-bound, reasoning-pattern skills transfer cleanly) are catalogued in Appendix A.3, and per-VLM seed re-evolution is the natural path forward.

Table 8. Memory ingest and retrieval rates per bench, Gemini full-inject `FullEvo`. Retrievals are evolver-fusion fetches; in this configuration memory is never injected into the per-question VLM prompt.

Bench	Ingests	Retrievals	Retrieval/Q	Avg units
EgoSchema	340	11	0.022	3.0
Video-MME long	578	22	0.024	3.0
EgoPlan	266	44	0.048	3.0
NextQA	745	17	0.017	3.0

Bank Composition and Memory Dynamics. The evolved bank grows substantially per run (28–55 new skills on a 12-seed base; Table 7), while memory retrieval fires only

~1× per 21–59 questions (Table 8) — a low-frequency, high-precision conditioning signal for the evolver. Notably, F1 (token-Jaccard dedup) and F2 (utility-tracking prune) remain inactive across all four benches at this run length (zero rejections, zero prunes; Appendix A.4), suggesting the bank has not yet reached the saturation regime where dedup and utility prune become load-bearing; this validates F1/F2 as scalability scaffolding rather than performance-critical components at the scales evaluated. On V-MME long the 12 seed skills accumulate 219 activations at 73% while the top-3 evolved skills accumulate 314 activations at 71%, reinforcing the §4.2 finding that the seed bank carries most of the lift on long-form benches (per-bench numerics in Appendix A.5). The cosine gate filters memory ingest in step with plain accuracy (NextQA 74.5% down to EgoPlan 28.8%), and retrieval fires only when the skill evolver runs, so the evolver receives ~40× less memory traffic than a per-question concatenation would deliver.

Potentials for Edge Device Deployment. Streaming wearables are the binding use case: for example, a 1-hour AI-glasses session at 1 fps emits 3,600 frames, which under a frontier-VLM full-frame upload would translate into roughly 3.9M input tokens and a tail-latency budget incompatible with cellular variance. MetaSight's on-device cascade rejects ~98% of those frames before any radio is woken, so the same hour ships 5–20 uploads end-to-end — consistent with the per-question keyframe rates measured in Table 5 (1.13–5.41 KF/Q across the four benches). Combined with `FullEvo`'s skill-driven accuracy lift (§4.2), the cascade therefore makes a class of AI-glasses video-QA agents viable that would otherwise be either too expensive (full-frame) or too inaccurate (uniform-8 plain) to deploy on a mobile data plan.

4.5 Case studies

Figure 2 traces two representative MetaSight wins on Gemini 3 Flash, one per recovery pathway. Both share the similar efficient encoding signature: the cascade compresses the clip down to a single MAJOR keyframe at $t \approx 1$ s (1/180 and 1/36), so the swing happens purely at the language layer. In Case 1, the baseline locks onto a surface noun on the keyframe ("picks paint") and skips the question's purpose-clause; the evolved skills inject a more concrete scope discipline at retrieval time and recovers GT **(D)** "loosens color intensity." Case 2 has no salient skill on retrieval — the lift instead comes from the memory bank: two prior confidence-gated exemplars on the "hold-tight → stabilise" pattern (rope-while-descending, handlebars-on-trail) re-route the evolver away from **(C)** "moving themselves" to GT **(B)** "hold for balance,". Two further wins of our approach, including a cross-VLM transfer to GPT-5.2, are presented in Appendix A.8.

5 Conclusion

We presented MetaSight, a self-evolving multimodal agent for cost-sensitive video question answering. The key idea is hybrid encoding: cheaply filter streaming frames before VLM upload, route only top-k skills into each prompt while keeping the rest in a cold catalogue, and use retrieved memories to evolve the skill bank offline rather than inflating every per-question call. Across four video-QA benchmarks and two VLM families, MetaSight substantially reduces API cost while improving accuracy in most settings, showing that multimodal agents can become both cheaper and stronger without weight updates. These results suggest a practical path toward deployable self-evolving video agents for live edge scenarios such as AI glasses, where streaming efficiency and continual adaptation are both essential.

References

[1] Anthropic. 2024. The Claude Model Family. https://www.anthropic.com/claude. Model overview page.

[2] Anthropic. 2024. Introducing the Model Context Protocol. https://www.anthropic.com/news/model-context-protocol. Open standard, specification at https://modelcontextprotocol.io.

[3] Anthropic. 2025. Introducing Claude Haiku 4.5. https://www.anthropic.com/news/claude-haiku-4-5. System card: https://anthropic.com/claude-haiku-4-5-system-card.

[4] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019. Efficient Lifelong Learning with A-GEM. In *International Conference on Learning Representations (ICLR)*. arXiv:1812.00420.

[5] Yi Chen, Yuying Ge, Yixiao Ge, Mingyu Ding, Bohao Li, Rui Wang, Ruifeng Xu, Ying Shan, and Xihui Liu. 2023. EgoPlan-Bench: Benchmarking Multimodal Large Language Models for Human-Level Planning. arXiv:2312.06722 [cs.CV]

[6] Sijie Cheng, Kechen Fang, Yangyang Yu, Sicheng Zhou, Bohao Li, Ye Tian, Tingguang Li, Lei Han, and Yang Liu. 2024. VidEgoThink: Assessing Egocentric Video Understanding Capabilities for Embodied AI. arXiv:2410.11623 [cs.CV]

[7] Sijie Cheng, Zhicheng Guo, Jingwen Wu, Kechen Fang, Peng Li, Huaping Liu, and Yang Liu. 2024. EgoThink: Evaluating First-Person Perspective Thinking Capability of Vision-Language Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2311.15596.

[8] Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building Production-Ready AI Agents with Scalable Long-Term Memory. arXiv:2504.19413 [cs.CL]

[9] Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. 2016. RL^2: Fast Reinforcement Learning via Slow Reinforcement Learning. arXiv:1611.02779 [cs.AI]

[10] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, Peixian Chen, Yanwei Li, Shaohui Lin, Sirui Zhao, Ke Li, Tong Xu, Xiawu Zheng, Enhong Chen, Caifeng Shan, Ran He, and Xing Sun. 2024. Video-MME: The First-Ever Comprehensive Evaluation Benchmark of Multi-modal LLMs in Video Analysis. arXiv:2405.21075 [cs.CV]

[11] Gemini Team, Google. 2024. Gemini 1.5: Unlocking Multimodal Understanding Across Millions of Tokens of Context. arXiv:2403.05530 [cs.CL]

[12] Google DeepMind. 2025. Gemini 3 Flash: Frontier Intelligence Built for Speed. https://deepmind.google/blog/gemini-3-flash-frontier-intelligence-built-for-speed/. Model release blog post; see also https:

//blog.google/technology/developers/build-with-gemini-3-flash/.

[13] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh Kumar Ramakrishnan, Fiona Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Zhongcong Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Abrham Gebreselasie, Cristina González, James Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Khoo, Jáchym Kolář, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Ziwei Zhao, Yunyi Zhu, Pablo Arbeláez, David Crandall, Dima Damen, Giovanni Maria Farinella, Christian Fuegen, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. 2022. Ego4D: Around the World in 3,000 Hours of Egocentric Video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2110.07058.

[14] Bo He, Hengduo Li, Young Kyun Jang, Menglin Jia, Xuefei Cao, Ashish Shah, Abhinav Shrivastava, and Ser-Nam Lim. 2024. MA-LMM: Memory-Augmented Large Multimodal Model for Long-Term Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2404.05726.

[15] Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. 2024. VTimeLLM: Empower LLM to Grasp Video Moments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2311.18445.

[16] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming Catastrophic Forgetting in Neural Networks. *Proceedings of the National Academy of Sciences (PNAS)* 114, 13 (2017), 3521–3526. arXiv:1612.00796.

[17] Jiapeng Li, Ping Wei, Wenjuan Han, and Lifeng Fan. 2023. IntentQA: Context-aware Video Intent Reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 11963–11974.

[18] Karttikeya Mangalam, Raiymbek Akshulakov, and Jitendra Malik. 2023. EgoSchema: A Diagnostic Benchmark for Very Long-form Video Language Understanding. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*. arXiv:2308.09126.

[19] OpenAI. 2025. GPT-5 System Card. https://openai.com/index/gpt-5-system-card/.

[20] OpenAI. 2025. Update to GPT-5 System Card: GPT-5.2. https://openai.com/index/gpt-5-system-card-update-gpt-5-2/. Model card update, December 11, 2025.

[21] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. 2023. MemGPT: Towards LLMs as Operating Systems. arXiv:2310.08560 [cs.AI]

[22] Pinelopi Papalampidi, Skanda Koppula, Shreya Pathak, Justin Chiu, Joe Heyward, Viorica Patraucean, Jiajun Shen, Antoine Miech, Andrew Zisserman, and Aida Nematzadeh. 2024. A Simple Recipe for Contrastively Pre-training Video-First Encoders Beyond 16 Frames. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2312.07395; introduces LongViViT.

[23] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (UIST)*. arXiv:2304.03442.

[24] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. 2019. Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*. arXiv:1903.08254.

[25] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 3982–3992. arXiv:1908.10084; underlying paper for the all-MiniLM-L6-v2 sentence-transformers model.

[26] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. 2024. TimeChat: A Time-sensitive Multimodal Large Language Model for Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2312.02051.

[27] Jonas Rothfuss, Dennis Lee, Ignasi Clavera, Tamim Asfour, and Pieter Abbeel. 2019. ProMP: Proximal Meta-Policy Search. In *International Conference on Learning Representations (ICLR)*. arXiv:1810.06784.

[28] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Advances in Neural Information Processing Systems (NeurIPS)*. arXiv:2302.04761.

[29] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. arXiv:2303.11366.

[30] Gunnar A. Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. 2018. Actor and Observer: Joint Modeling of First and Third-Person Videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:1804.09627; introduces Charades-Ego.

[31] Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, Yan Lu, Jenq-Neng Hwang, and Gaoang Wang. 2024. MovieChat: From Dense Token to Sparse Memory for Long Video Understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2307.16449.

[32] Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, Peng Xia, Fang Wu, He Zhu, Ge Zhang, Jiaheng Liu, Xingyao Wang, Sirui Hong, Chenglin Wu, Hao Cheng, Chi Wang, and Wangchunshu Zhou. 2025. Agent KB: Leveraging Cross-Domain Experience for Agentic Problem Solving. arXiv:2507.06229 [cs.AI]

[33] Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024. Voyager: An Open-Ended Embodied Agent with Large Language Models. *Transactions on Machine Learning Research* (2024). arXiv:2305.16291.

[34] Xiaohan Wang, Yuhui Zhang, Orr Zohar, and Serena Yeung-Levy. 2024. VideoAgent: Long-form Video Understanding with Large Language Model as Agent. In *Proceedings of the European Conference on Computer Vision (ECCV)*. arXiv:2403.10517.

[35] Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. 2025. VideoTree: Adaptive Tree-based Video Representation for LLM Reasoning on Long Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2405.19209.

[36] Peng Xia, Jianwen Chen, Xinyu Yang, Haoqin Tu, Jiaqi Liu, Kaiwen Xiong, Siwei Han, Shi Qiu, Haonian Ji, Yuyin Zhou, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. 2026. MetaClaw: Just Talk – An Agent That Meta-Learns and Evolves in the Wild. arXiv:2603.17187 [cs.AI] Preprint; arXiv ID 2603.17187 from local PDF metadata. Verify on arXiv after public listing..

[37] Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. 2021. NExT-QA: Next Phase of Question-Answering to Explaining Temporal Actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 9777–9786. arXiv:2105.08276.

[38] Antoine Yang, Arsha Nagrani, Paul Hongsuck Seo, Antoine Miech, Jordi Pont-Tuset, Ivan Laptev, Josef Sivic, and Cordelia Schmid. 2023. Vid2Seq: Large-Scale Pretraining of a Visual Language Model for Dense Video Captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. arXiv:2302.14115.

[39] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. 2023. Self-Chained Image-Language Model for Video Localization and Question Answering. In *Advances in Neural Information Processing Systems (NeurIPS)*. arXiv:2305.06988.

[40] Sicheng Yu, Chengkai Jin, Huanyu Wang, Zhenghao Chen, Sheng Jin, Zhongrong Zuo, Xiaolei Xu, Zhenbang Sun, Bingni Zhang, Jiawei Wu, Hao Zhang, and Qianru Sun. 2025. Frame-Voyager: Learning to Query Frames for Video Large Language Models. In *The Thirteenth International Conference on Learning Representations (ICLR)*. arXiv:2410.03226.

[41] Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual Learning Through Synaptic Intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*. 3987–3995. arXiv:1703.04200.

[42] Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. 2024. A Simple LLM Framework for Long-Range Video Question-Answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. arXiv:2312.17235.

[43] Guibin Zhang, Haotian Ren, Chong Zhan, Zhenhong Zhou, Junhao Wang, He Zhu, Wangchunshu Zhou, and Shuicheng Yan. 2025. MemEvolve: Meta-Evolution of Agent Memory Systems. arXiv:2512.18746 [cs.AI]

[44] Hang Zhang, Xin Li, and Lidong Bing. 2023. Video-LLaMA: An Instruction-tuned Audio-Visual Language Model for Video Understanding. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (EMNLP Demo)*. arXiv:2306.02858.

[45] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. 2025. Video Instruction Tuning With Synthetic Data. *Transactions on Machine Learning Research* (2025). arXiv:2410.02713; introduces LLaVA-Video.

[46] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. 2024. ExpeL: LLM Agents Are Experiential Learners. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*. arXiv:2308.10144.

A Appendix

A.1 Compute cost

Total experiments reported in this paper: ~$190 across Gemini 3 Flash and GPT-5.2 (Azure proxy). Reasoning-token billing on the proxy is not separately exposed and may inflate the GPT-5.2 portion by 1.5–3× depending on Azure deployment configuration.

A.2 Cascade-fill on long-clip and uniform-activity benches

Table 6 reports parity-budget ($K = 8$) cascade-fill on the two short-clip benches where the hybrid wins; we list here the same comparison on EgoSchema (3-min clips, uniform activity) and Video-MME long (30+ min clips). Per-bench `min_gap_s`: EgoSchema 5 s, V-MME long 60 s, EgoPlan 1 s, NextQA 1 s; all runs use `max-keyframes=8`. GPT-5.2 cascade-fill experiments are not yet measured.

The bench-conditional split reflects the underlying frame-relevance distribution. On EgoSchema's 3-min uniform-activity ego clips, salient content is distributed roughly evenly through the clip rather than concentrated in scene transitions, so the cascade gate's content-aware selection has less leverage and uniform-8's even temporal coverage is harder to beat. On V-MME long's 30+ min clips the situation is different but converges to the same conclusion: with `min_gap_s` forced to 60 s to keep the cascade-fill samples non-overlapping, the fill positions are too coarse to recover the content the cascade missed in the long static stretches between scene transitions. We report cascade-fill as a parity-budget alternative that is bench-conditional in its win, not a universal replacement for uniform-8.

A.3 Cross-VLM transfer dynamics

Cross-VLM transfer dynamics are per-skill rather than bank-wide: format-enforcement skills are VLM-family-bound while reasoning-pattern skills transfer cleanly, and the memory-injection mode preference can invert sign across VLM families. A targeted ablation on the GPT-5.2 EgoSchema row of Table 2 surfaces three findings.

(i) Format-enforcement is VLM-family-bound. Removing the single `answer-format-completion` skill from the seed bank (`seed-11`) recovers GPT-5.2 by +4 to +6 % uniformly across three benches at n=50, because the skill was evolved against a Gemini failure mode (premature abandonment) that GPT-5.2 does not exhibit. *Caveat:* promoting `seed-11` as a universal bank fails for Gemini at full bench (−4.2 % regression on EgoSchema 500), so per-VLM bank composition is necessary.

(ii) Reasoning-pattern skills transfer cleanly. The remaining 11 reasoning-pattern skills deliver +2.60 % on GPT-5.2 EgoSchema even without the format skill, mirroring the Gemini "seed alone is most of the lift" pattern.

Table 9. Approximate compute costs by stage. "Wall" = approximate wall-clock with 6–12 concurrent runs. All Bedrock Haiku evolver calls are bundled into the Gemini-side accounting.

Stage	Runs	Approx. cost (USD)	Approx. wall (h)
Tier-1 ablation ladder (EgoSchema/V-MME short/TeleEgo)	16	46	18
Tier-2 cascade per-stage breakdown	5	2	1
Cross-VLM full bench (GPT-5.2 6 runs)	6	36	8
Headline 4-bench × 2-VLM experiments	8	∼50	∼12
NextQA + EgoPlan ablation ladders	14	∼50	∼10
Total	∼50	∼**190**	∼50

(iii) Memory-mode preference inverts. `+SkillMemCat` costs −2.0 % on Gemini EgoSchema but delivers +3.20 % on GPT-5.2 EgoSchema, while memory→evolver fusion is positive on both VLMs — so direct concatenation is bench/VLM-conditional rather than universally hurtful, and we keep `FullEvo` as the universal recommendation. Combined with the V-MME long picture (full-inject regresses on GPT-5.2; hybrid-3 recovers), the rule is: method transfers cleanly on the evolution-source bench, while non-source benches require per-VLM injection-mode tuning.

A.4 Bank hygiene activity on supplementary benches

The F1 (token-Jaccard dedup at evolve-time) and F2 (per-skill utility prune) filters fired zero times on the four headline benches: at 500–1000 questions per run the Haiku evolver's name-generation diversity is sufficient to avoid near-duplicates, and per-skill hit rates stay within 5 % of the bank mean. On the longer / more diverse-task supplementary benches (V-MME short, TeleEgo) the filters do fire — F1 logs 11 and 5 rejections respectively, F2 fires 2 prune events (10 skills dropped) — delivering a 24–37 % bank-size reduction at ± 1 % accuracy. F1+F2 are therefore best understood as low-cost insurance for longer evolution histories rather than active levers in the headline configuration.

A.5 Per-skill EgoPlan numerics

On EgoPlan, all top-tier seed skills bottom out near 27 % accuracy (225 activations each), mirroring the bench's near-random absolute ceiling for non-frontier VLMs that lack visible task-progress context (§4.4). Top evolved skills accumulate 204 activations at 26 %; later-evolved entries accumulate 148–188 activations at 28–29 %. The per-skill spread is below the ablation noise floor on this bench, which is consistent with the headline finding that EgoPlan is at near-random absolute accuracy regardless of method.

A.6 Cascade per-stage breakdown

We retain the per-stage breakdown for transparency, despite production `cg-adaptive` losing offline accuracy on EgoSchema. Table 10 (plain Gemini 3 Flash, EgoSchema 200, seed=42, $\tau_{\text{major}} = 0.20$) shows three things: (i) uniform-8 wins offline by +9.5 % at 8 KF/Q on this bench; (ii) `dhash-only` is the worst cascade mode because hash-passed frames cluster toward early video, so deduplication alone without scene-aware reweighting underperforms even uniform sampling; (iii) adaptive thresholds and temporal decay cost ∼4 % relative to static thresholds at matched KF/Q on EgoSchema. The adaptive features are the only mode robust to live-streaming conditions (slow-moving scenes, stationary cameras, irregular frame arrival), and the cost reverses on diverse-scene benches: `cg-adaptive` wins `cg-static` by +3.4 % on TeleEgo. We default to `cg-adaptive` because cross-bench portability under streaming conditions outweighs the EgoSchema-specific offline gap.

Table 10. Cascade per-stage breakdown on EgoSchema 200, plain Gemini 3 Flash, seed=42, $\tau_{\text{major}} = 0.20$. KF/Q = avg keyframes per question sent to the VLM.

Mode	Pipeline	Acc. (%)	KF/Q	Input tokens
`uniform-8`	uniform sampling, no gate	**66.0**	8.00	1,761,645
`dhash-only`	dHash → take all hash-passed	57.0	8.00	1,761,645
`cg-static`	dHash + LE + CG (no decay)	60.5	4.83	1,077,301
`cg-adaptive`	full pipeline (production)	56.5	4.83	1,083,269

A.7 Full per-experiment token / cost / latency profile

The headline saving in §4.3 (Table 5) compares Gemini Cascade + `FullEvo` against the Uniform-8 + `FullEvo` offline ceiling. Table 11 reports the full per-experiment profile for both VLM families across the four benchmarks plus the cross-VLM ablation runs, drawn from the `vlm_usage` field of each results dump.

A.8 Additional case studies

Table 11. Per-experiment token / cost / latency profile across the headline 4-bench × 2-VLM experiments plus key cross-VLM ablation runs. $/Q assumes Gemini 3 Flash at $0.30/M input + $2.50/M output and GPT-5.2 (Azure proxy) at $1.25/M input + $10/M output; reasoning-token billing on the proxy is not separately exposed and may inflate the GPT-5.2 cost by 1.5–3×. KF/Q = avg keyframes per question after the cascade.

Bench	Variant	Acc. (%)	in_tok/Q	out_tok/Q	$/run	$/Q	lat./Q (s)	KF/Q
EgoSchema	Gemini plain	52.60	3,430	15.1	$0.53	$0.0011	9.05	2.96
	Gemini FullEvo	68.00	9,524	5.9	$1.44	$0.0029	9.80	2.95
	GPT-5.2 plain	64.00	905	5.0	$0.14	$0.0003	4.39	2.95
	GPT-5.2 FullEvo	68.00	6,369	4.9	$0.96	$0.0019	5.43	2.95
V-MME long	Gemini plain	60.33	6,111	10.8	$1.67	$0.0019	14.56	5.43
	Gemini FullEvo	64.22	13,420	2.3	$3.63	$0.0040	13.66	5.41
	GPT-5.2 plain	55.89	5,461	4.9	$6.18	$0.0069	9.82	5.40
	GPT-5.2 FullEvo	55.89	6,307	3.8	$7.13	$0.0079	12.58	5.40
EgoPlan	Gemini plain	24.62	1,352	10.5	$0.40	$0.0004	8.55	1.13
	Gemini FullEvo	28.85	10,728	1.5	$2.97	$0.0032	8.99	1.13
	GPT-5.2 plain	28.42	1,198	5.1	$0.34	$0.0004	5.43	1.13
	GPT-5.2 FullEvo	28.85	10,207	5.2	$2.83	$0.0031	4.11	1.13
NextQA	Gemini plain	72.70	1,750	4.0	$0.53	$0.0005	5.70	1.51
	Gemini FullEvo	74.50	8,207	1.3	$2.47	$0.0025	5.84	1.51
	GPT-5.2 plain	73.20	576	5.1	$0.18	$0.0002	2.95	1.51
	GPT-5.2 hybrid-3	68.10	2,244	4.8	$0.68	$0.0007	9.24	1.51

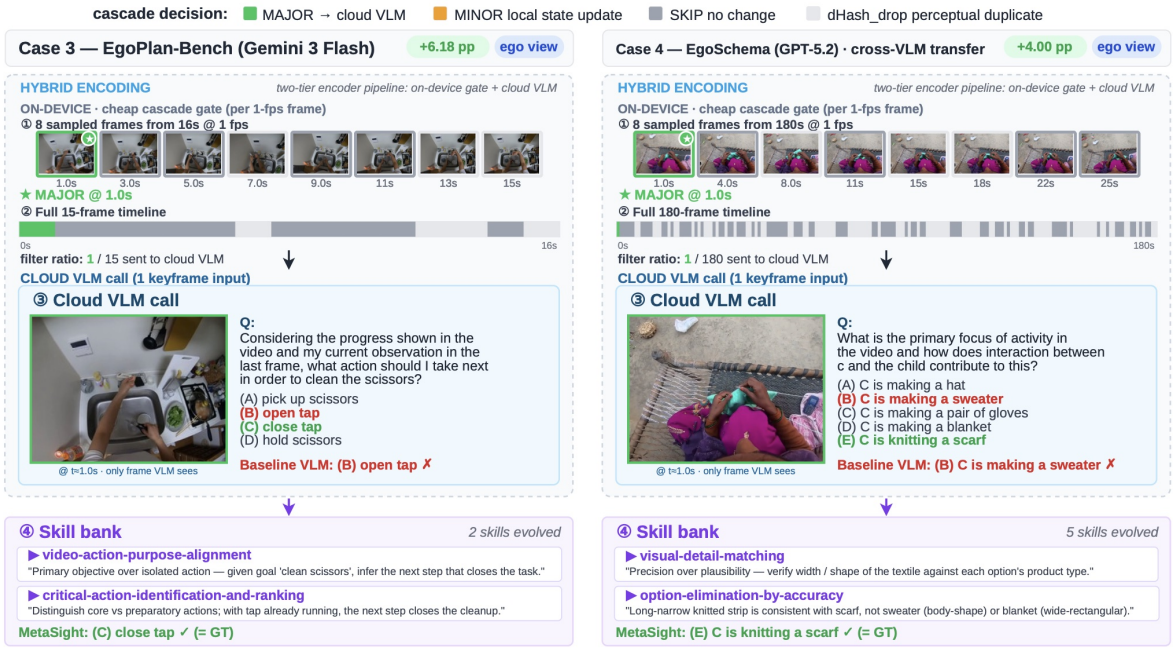

Figure 3. Two further MetaSight wins, including a cross-VLM transfer. **Case 3** (EgoPlan-Bench, Gemini 3 Flash, **+6.18**%): single keyframe (1/15); evolved skills flip **(B)** "open tap" → GT **(C)** "close tap." **Case 4** (EgoSchema, GPT-5.2, **+4.00**%, cross-VLM): a Gemini-evolved bank applied unmodified to GPT-5.2 corrects **(B)** "making a sweater" → GT **(E)** "knitting a scarf."